uc3m | Universidad Carlos III de Madrid

Grado Universitario en Ingeniería Electrónica Industrial y Automática

2017-2018

Trabajo Fin de Grado

"Diseño de interfaz hombre-máquina para vehículo autónomo"

María Huertas Martín

Tutor

Fernando García Fernández

Leganés, 2018

RESUMEN

En la actualidad, se puede considerar que la implicación en el desarrollo del vehículo autónomo se encuentra en auge. Multitud de compañías invierten tiempo y dinero en investigar cuáles son las mejores vías para la inclusión de este tipo de vehículos en la sociedad.

El objetivo principal de este proyecto es el de estudiar las limitaciones sociales que tiene un peatón mientras desarrolla una acción tan rutinaria como sería cruzar un paso de peatones sin semáforo y por donde se encuentra a su vez circulando un vehículo sin conductor Para esto, se decide crear una interfaz capaz de crear una comunicación entre vehículo autónomo y peatón, aportándole así cierta confianza para realizar la acción planteada anteriormente.

La interfaz que se va a realizar se basa en el muestreo de imágenes, a través de una pantalla instalada en el iCab 1, pasando por dos estados, detectado o no detectado. Lo que se espera por parte del peatón es que sea capaz de interpretar los mensajes emitidos y así crear esa cierta comunicación entre ambos.

Para el desarrollo de la interfaz se creará un paquete utilizando GNU/Linux junto con ROS, un software orientado a robots que se encuentra previamente instalado en el vehículo iCab1. Como lenguaje de desarrollo de código se ha utilizado C++, junto con las librerías de OpenCV, necesarias para mostrar por pantalla las imágenes de estados del peatón.

Para concluir con el proyecto, se han realizado una serie de experimentos reales en la universidad Carlos III de Madrid para analizar posteriormente los resultados mediante una serie de cuestionarios entregados a cada uno de los participantes.

Palabras clave: vehículo autónomo; interfaz; comunicación; peatón; ROS

ÍNDICE DE CONTENIDOS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

ACRÓNIMOS

ADAS Advanced Driver-Assistance Systems

BOE Boletín Oficial del Estado

DGT Dirección General de Tráfico

GPS Global Positioning System

iCab Intelligent Campus automobile

IMU Inertial Measurement Unit

ITS Intelligent Transport System

IvvI Intelligent vehicle based on visual Information

LIDAR LIght Detection And Ranging

LSI Laboratorio de Sistemas Inteligentes

Qt Q toolkit

ROS Robot Operating System

UC3M Universidad Carlos III de Madrid

VAs Vehículos Autónomos

1. INTRODUCCIÓN

1.1. Necesidades que cubrir / propósito del proyecto

La existencia de vehículos autónomos es ya una realidad y gracias a las numerosas investigaciones que se están realizando, cada vez están más presentes en nuestro día a día. Es por ello por lo que surge la necesidad de crear interfaces que faciliten su uso a todas las personas que vayan a interactuar con ellos, desde los conductores hasta los peatones.

El propósito principal de este proyecto es el de mejorar el comportamiento, que se da en la actualidad, cuando un peatón tiene la necesidad de cruzar una vía y se encuentra ante un vehículo sin conductor, estudiando cuáles serán sus posibles reacciones. En primer lugar, las decisiones que toma el peatón pueden causar situaciones de incertidumbre, lo cual puede provocar que el vehículo autónomo, en el peor de los casos, no consiga detenerse a tiempo. Para evitar esto, surgió la necesidad de crear un lenguaje de comunicación donde el vehículo autónomo pueda avisar al peatón que fue detectado y es completamente seguro cruzar, y así no crear el conflicto anteriormente mencionado.

Hoy en día son muchas las empresas que se han planteado la creación de esta interfaz y muchas de ellas ya las están llevando a cabo, por lo que se está considerando realmente importante la figura del peatón, no solo por la necesidad de crear seguridad en él, sino también para evitar accidentes.

Para alcanzar el propósito principal nombrado anteriormente, es importante conocer la opinión del peatón ante la incertidumbre de si el vehículo que se aproxima es realmente un vehículo autónomo. Por eso, en el presente proyecto se realizarán y analizarán una serie de experimentos reales para conocer así la opinión del peatón ante dicha interfaz en desarrollo, y poder aplicarla de la mejor forma en el futuro.

1.2. Objetivos que desarrollar

El objetivo principal de este proyecto es el de desarrollar una interfaz que facilite la comunicación entre un vehículo autónomo por tierra y un peatón que tiene la necesidad de cruzar una vía donde existe un paso de peatón sin semáforo.

A continuación, se van a describir los objetivos específicos que se tienen que lograr previos al objetivo final:

- Recoger información hasta la fecha sobre los vehículos autónomos y su forma de comunicación con el peatón que se aproxima, para poder aplicarla en el proyecto a tratar.
- Adquisición de conocimientos sobre ROS, para poder aplicarlos en el desarrollo del software. Estos incluyen todo el proceso de aprendizaje hasta la creación del paquete final, el cual interactúa con el resto de los paquetes ya existentes en el vehículo autónomo.
- Búsqueda y aprendizaje de librerías en C++ para la integración con ROS en la visualización de imágenes útiles para la comunicación con el peatón.
- Adquisición de conocimientos en Bitbucket con uso de Git, como software de control de versiones.
- Creación de encuestas de valoración, las cuales nos darán la respuesta de los peatones ante la aproximación de un vehículo autónomo.
- Posterior análisis de las pruebas. Esto nos otorgara una opinión real del proyecto desarrollado y las posibles mejoras posteriores que se pueden llegar a realizar en caso de necesitar mejorar el sistema.

2. ESTADO DEL ARTE

2.1. Introducción al concepto de vehículos autónomos

Se dice que un vehículo autónomo es aquel capaz de combinar numerosas técnicas, englobadas en su hardware y software, para poder moverse sin precisar de la ayuda de un conductor. Los elementos que forman el hardware son los capaces de percibir el entorno, mientras que el software está formado por los complejos algoritmos que reciben dicha información y emiten una decisión.

Para ello, es importante que el vehículo autónomo controle en todo momento el entorno que le rodea. Podemos resumir estas tareas en cuatro bloques [1]:

- Localización: debe saber dónde se encuentra.
- Presencias en el entorno: esto va desde coches y personas hasta posibles objetos y señales. Además, debe saber interpretar cada una de ellas, evitando así que el VA tenga dudas de código y se cree ambigüedad.
- Ruta que realizar: el VA debe conocer cuál es el trayecto asignado. Esto incluye el punto de salida, el punto de llegada y el recorrido entre ambos puntos.
- Estado del conductor: como se describe dentro del punto 2.2., existen dos niveles de autonomía que incluyen los VA. Dentro del nivel 4, aunque la mayor parte del vehículo es automatizada, aún se requiere de la existencia de un conductor, por lo que es sumamente importante conocer el estado. El VA debe conocer si se encuentra descansando, despistado, o por el contrario está disponible para asumir cualquier maniobra del vehículo tomando así el control.

Para entender el proceso de los vehículos autónomos es importante conocer cómo se han ido desarrollando a lo largo de la historia. Lo cierto es que este proceso no ha sido repentino, ya que en 1925 Francis P. Houdina creó el primer automóvil controlado por radio y lo mostró al mundo. Según el New York Times el vehículo era capaz de cambiar de marchas, encender el motor y pitar [2, p. 20].

592 *Radio News for November, 1925*

Radio-Controlled Automobile

By HERNDON GREEN

Radio is to control a car in transcontinental tour. The system, which is extremely simple and effective, is fully described here.

Above is shown the two cars used in the transcontinental tour by Mr. Francis P. Houdina. The front car has no driver but is controlled by radio from the second car. Note the transmitting antenna and the receiving loop.

THE DIAGRAM

A glance at the wiring diagram given in an accompanying illustration will show the circuit in its simplest form. Some of the amplifying relays are left out in order to simplify the circuits.

The two transmitters are of the usual 10-watt type, using storage battery supply for the filaments and plates. They are housed in the tonneau of the control car together with the power units. The keys controlling them are placed on a small shelf at the right side of the dash.

The receivers are the usual type, employ-

Ilustración 2.1: Primer automóvil controlado por radio [3]

A partir de ese hecho, se incentivó la curiosidad por la creación de VA. En 1939 se presentó en la feria de muestras Futurama el primer vehículo eléctrico controlado mediante un circuito eléctrico colocado en el pavimento [2, p. 22].

No fue hasta 1980 cuando se introducen nuevas técnicas, que incluso a día de hoy se siguen utilizando y mejorando. En ese año, DARPA presenta el primer vehículo que utiliza radares láser y visión por ordenador.

En 1990, Dean Pomerleau escribe su tesis doctoral e incluye en ella el concepto de redes neuronales. Las redes neuronales son las que permiten a los sensores captar imágenes y enviar órdenes en tiempo real al controlador del vehículo autónomo. Y sería en 1995 cuando Pomerleau y Todd Jochem llevarían a cabo su propio sistema automatizado [4].

Pero realmente fue a partir de 1994 cuando pudimos ver a dos coches, VaMP y Vita-2, de Daimler-Benz y Ernst Dickmans, conducir de manera autónoma con pequeñas intervenciones humanas a lo largo de más de 1000 kilómetros de autovía formada por 3 carriles en París, alcanzando velocidades de hasta 130Km/h [5]. En este experimento se llegó a demostrar que era posible la conducción autónoma y el cambio autónomo de carril en una carretera formada por un tráfico constante de vehículos no autónomos.

Otro hecho relevante tuvo lugar en 1995, cuando Dickmanns incorporó mejoras a un Mercedes para que pudiera realizar un viaje usando visión computarizada y un ordenador con un gran procesador que pudiera mandar órdenes en tiempo real. En algunos periodos de tiempo se alcanzaron velocidades superiores a 175Km/h, con intervenciones

humanas en únicamente un 5% del trayecto (ida y vuelta Munich-Copenague), es decir, el 95% fue conducción autónoma. Además, se realizaron adelantamientos de hasta 110 Km/h [2, p. 45].

Todos estos hechos marcaron un antes y un después en la conducción autónoma. Gracias a ellos, Google, en 2009, decide empezar un proyecto secreto que no fue descubierto hasta 2015, cuando presenta públicamente su prototipo vehículo autónomo sin conductor, volante, acelerador ni freno. Este vehículo tomó el nombre de Waymo [6].

Ilustración 2.2: Prototipo de vehículo autónomo presentado por Google [7]

Y a partir de 2013 otras empresas de automóviles se unen, creando proyectos independientes, y mostrándose como competidoras entre ellas.

En la actualidad, se sigue trabajando en la incorporación de nuevas mejoras que, además, puedan reducir el coste del vehículo al futuro consumidor.

La principal meta que alcanzar es encontrar el vehículo autónomo perfecto, es decir, queremos lograr el sistema de transporte más seguro hasta la fecha cuyo funcionamiento se base en recibir señales y emitir ordenes sin posibilidad a error.

Esto último es realmente importante y crea mucha controversia a la par, debido a que la prudencia y control que un algoritmo puede aportar es insuperable por el ser humano, pero la reacción de un ser humano ante un posible accidente es imposible inculcársela a una máquina. Es aquí donde entra en juego la ética del vehículo autónomo, la cual es otra de las metas a superar, extensa y difícil de abordar.

Dentro de las futuras mejores que se están llevando a cabo en la actualidad es la que se presenta en este proyecto, es decir, las empresas están invirtiendo dinero en crear un método de comunicación entre un peatón y el vehículo autónomo eficaz.

2.2. Niveles de autonomía de los vehículos

Para poder clasificar los vehículos en función de sus características de autonomía del conductor, se han definido 6 niveles [8]. Desde el nivel 0 donde se encuentran aquellos vehículos que carecen de sistemas que ayuden al conductor, hasta el nivel 5 estarían los vehículos totalmente automatizados y que carecen de conductor.

A continuación, se van a describir cada uno de esos niveles y sus características [9].

- NIVEL 0: No automatizado: Este nivel engloba a todos aquellos vehículos donde el conductor no recibe ningún tipo de ayuda del coche, es decir, tiene el 100% del control de este. Todas las entradas que recibe el coche son por parte del conductor.

- NIVEL 1: Asistencia al conductor: En este nivel, aunque el conductor es el que maneja el vehículo, sí que puede recibir ayudas por parte del coche. Un ejemplo podría ser el control activo de crucero, donde el conductor le indica una velocidad al vehículo y este la asume, pudiendo además modificarla en caso de que se encuentre con un automóvil delante.

  En este nivel, los vehículos pueden incorporar sensores como ultrasonidos y cámaras.

- NIVEL 2: Conducción parcialmente automatizada: En este nivel el conductor deberá tanto seguir permaneciendo atento a la carretera como ser capaz de asumir el control cuando así sea necesario, pero el vehículo empieza a formar parte de la toma de decisiones.

  Combinadas las funciones de este nivel con las funciones del nivel 1, el vehículo podrá automáticamente dirigir, acelerar y frenar en situaciones limitadas.

  Algunos ejemplos de características pertenecientes a este nivel serían el asistente de dirección y control de carril o estacionamiento por control remoto.

- NIVEL 3: Conducción altamente automatizada: Este nivel incluye los vehículos que son capaces de conducir de manera autónoma, es decir, sin intervención del

conductor, durante largos periodos de tiempo. Estos periodos suelen ser en autovías, donde los vehículos no tienen interferencias de peatones o similares, como sería el caso de los núcleos urbanos.

Estos vehículos aún no se encuentran en el mercado actual, siguen en proceso de desarrollo. Algunas compañías, como es el caso de BMW estiman que su comercialización será viable a partir del año 2021.

- NIVEL 4: Conducción completamente automatizada: Como su propio nombre indica, en este nivel el vehículo es el que toma el control, hasta en núcleos urbanos con situaciones complejas de abordar. Sin embargo, sigue habiendo asiento para el conductor, y este puede tomar el control del vehículo cuando lo desee o si fuera necesario.

  Si se da una situación que cree ambigüedad, o bien que el propio vehículo no sepa cómo abordar, este le cederá el control al conductor. Si este, por alguna razón no se encuentra en posición para tomar el control, el vehículo podrá pasar a un estado de seguridad, que bien podría ser la parada del vehículo en una zona segura.

- NIVEL 5: Automatización completa: En este nivel, todas las personas que se encuentren dentro del vehículo son pasajeros. Es decir, no se necesita que ningún integrante este en posición de toma de control, ni si quiera es necesario que tenga licencia de conducir. Es posible que, además, por esos motivos, el coche carezca de cabina de conducción.

  Para cumplir con todos los requisitos de este nivel, el vehículo debe ser capaz de actuar en cualquier situación sin necesidad de ayuda humana. Esta meta aún está muy lejos de conseguirse, pero sería el nivel más alto que un vehículo podría llegar a tener por el momento.

  Inicialmente, para tener controladas todas las situaciones, en este nivel los vehículos estarían integrados en núcleos definidos de las ciudades, y alcanzarían velocidades bajas si dichos núcleos están poblados.

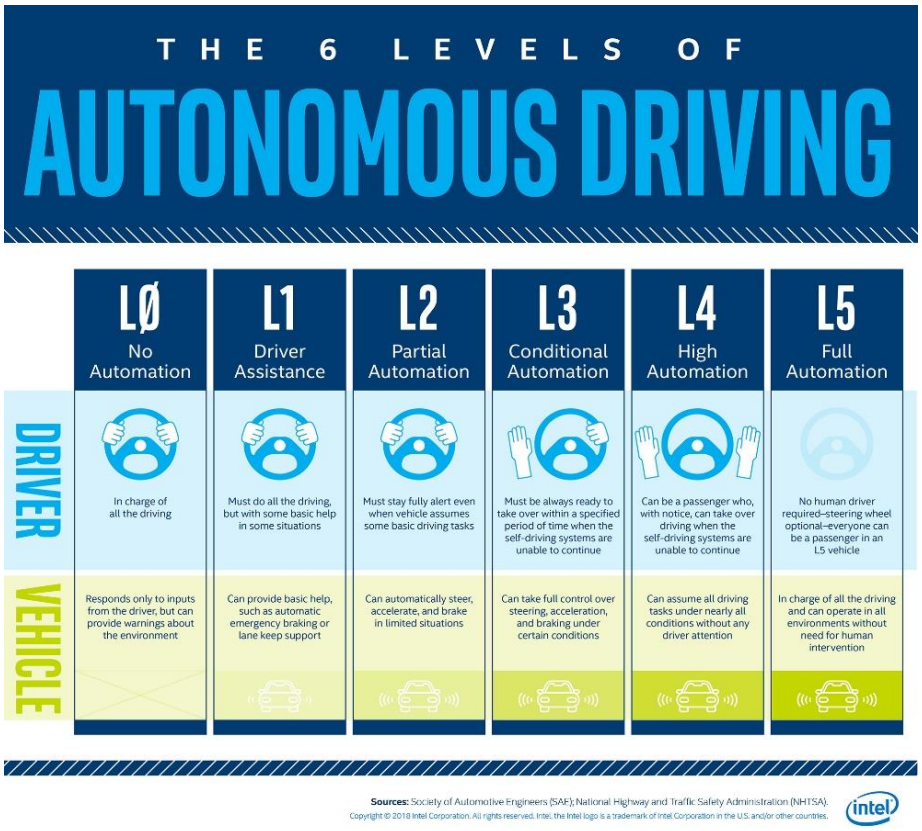


Ilustración 2.3: Características que debe cumplir cada vehículo dentro de los seis niveles de autonomía [10]

La anterior imagen recopila un resumen gráfico de las características que debe tener cada vehículo dentro de cada uno de los diferentes niveles de autonomía.

2.3. Hardware dentro del vehículo autónomo

El hardware del vehículo son los elementos físicos que lo componen. A continuación, se describirán cada uno de los elementos básicos necesarios para que un vehículo pueda funcionar de manera autónoma.

Cada uno de los elementos se encuentran comunicados con el software del vehículo, el cual es el encargado de recibir señales de los elementos emisores y de emitir órdenes a los elementos receptores, que actuarán de una u otra forma en función de estas decisiones.

Ilustración 2.4: Elementos que conforman un vehículo autónomo y su colocación [11]

Las partes más importantes de los vehículos autónomos son las siguientes [1]:

- Ordenador central: Es el encargado de procesar todas las órdenes que los sensores y demás elementos receptores reciben, para que el software junte y procese toda esa información, y la emita en forma de decisión final. Está formado por varios núcleos, los cuales aportan velocidad al sistema, ya que tiene que trabajar en tiempo real y debe emplear el menor tiempo posible en emitir las decisiones.

- Sensor Lidar: LIDAR es un acrónimo del inglés, Laser Imaging Detection and Ranger, que traducido al español se trataría de un láser que se utiliza para medir la distancia desde la parte emisora hasta el objeto en cuestión. Su funcionamiento consiste en una medición del tiempo que tarda el haz de luz emitido en llegar al objeto que tenemos delante.

  Los sensores LIDAR utilizados en vehículos autónomos tienen un rango de 360º, es decir, pueden trabajar en todas las direcciones. Además, otra de las ventajas de este tipo de sensores es que funcionan bajo cualquier condición meteorológica y de iluminación.

- Cámaras de visión: Como este tipo de vehículos no requieren de ayuda humana, se le colocan varias cámaras, según necesidades, que imitan de cierta manera la visión humana. Para ello, también se requiere un trabajo posterior de estas imágenes usando en gran medida técnicas de visión artificial.

- Radares: El uso de radares ultrasonidos es complementario al uso de sensores LIDAR, ya que la función principal es la misma, detectar distancias. Detectar distancias con otros vehículos puede ser muy útil, por ejemplo, para conocer la velocidad a la que circulan los demás y de esa manera poder tomar decisiones que les involucren.

- GPS/IMU: IMU, o unidad de medición inercial que, con ayuda de acelerómetros y giróscopos, es capaz de medir la orientación, velocidad y dirección de desplazamiento. La IMU es complementario al GPS, y se utiliza en caso de que este no tenga cobertura, o para contrastar resultados.

- Sensores sonoros: En algunos de los vehículos también puede haber micrófonos integrados, otorgando sentido auditivo al vehículo, y que junto con una serie de algoritmos que trabajen con la información recogida por estos, pueden ofrecer nuevas variables a la hora de tomar una decisión.

- Sensores y cámaras internas: Este tipo de sensores aparecen sobre todo en vehículos de nivel 3 y 4, donde estos pueden conducir de manera autónoma pero aún se requiere ayuda de un conductor en ciertas situaciones. Estos sensores o cámaras colocados en el interior del vehículo pueden monitorizar en todo momento el estado del conductor. Este tipo de sensores aportan información irrelevante a vehículos de nivel 5, ya que no son vehículos donde no existe ni se requiere un conductor.

Estos elementos del hardware, junto con el software, conformarían el funcionamiento de todo vehículo completamente autónomo.

2.4. Marco regulador

Como se ha mostrado en los puntos anteriores, el concepto de conducción autónoma lleva presente más de 60 años, y aunque su desarrollo real alrededor de 30 años, podemos observar y decir con seguridad que aún queda mucho por avanzar.

A día de hoy, el desarrollo tecnológico que suponen los vehículos autónomos va más deprisa que la creación de leyes que los protegen [12]. Con esto hago referencia a que actualmente solo existen dos leyes reguladoras, e inicios de proyecto de leyes aún sin aprobar.

Es por ello por lo que las limitaciones que hoy se encuentran para desarrollar los vehículos autónomos no son únicamente tecnológicas, sino que en mayor medida son marcadas por otro tipo de factores, como pueden ser políticos, la falta de infraestructuras donde ponerlos en funcionamiento públicamente, u otros tipos de problemas como es el de la responsabilidad jurídica y la ética [13].

También cabe mencionar que, la regulación de los vehículos autónomos está dividida, es decir, cada país tiene impuestas sus propias regulaciones.

Dentro de la Unión Europea, se aprobó la Directiva 2010/40/UE del Parlamento Europeo y del Consejo[1]. Dicho marco regulador aplicado en la UE quería crear un marco común para todos los países integrantes. Posteriormente, en España, se aprobó el Real Decreto 662/2012[2].

En general, España es considerado uno de los países donde más medidas se ponen para facilitar legalmente el desarrollo de los vehículos autónomos. Existe un marco legal desde 2015 por el que los fabricantes pueden probar sus prototipos de vehículos autónomos pidiendo un permiso especial a la DGT para que puedan circular.

En la actualidad en España, DGT y Mobileye, propiedad de Intel, están desarrollando leyes regulatorias que incrementen la seguridad vial y así poder permitir el avance en el desarrollo de los vehículos autónomos en nuestras carreteras [14]. Esto es debido, a que

[1] Directiva 2010/40/UE del Parlamento Europeo y del Consejo de 7 de julio de 2010, por la que se establece el marco para la implantación de los sistemas de transporte inteligentes en el sector del transporte por carretera y para las interfaces con otros modos de transporte.
[2] Real Decreto 662/2012, de 13 de abril, por el que se establece el marco para la implantación de los sistemas inteligentes de transporte (SIT) en el sector de transporte por carretera y para las interfaces con otros modos de transporte.

la DGT considera que el uso de ADAS permitirá disminuir la siniestralidad de los vehículos, así como las posibles muertes que esto puede llegar a suponer [15].

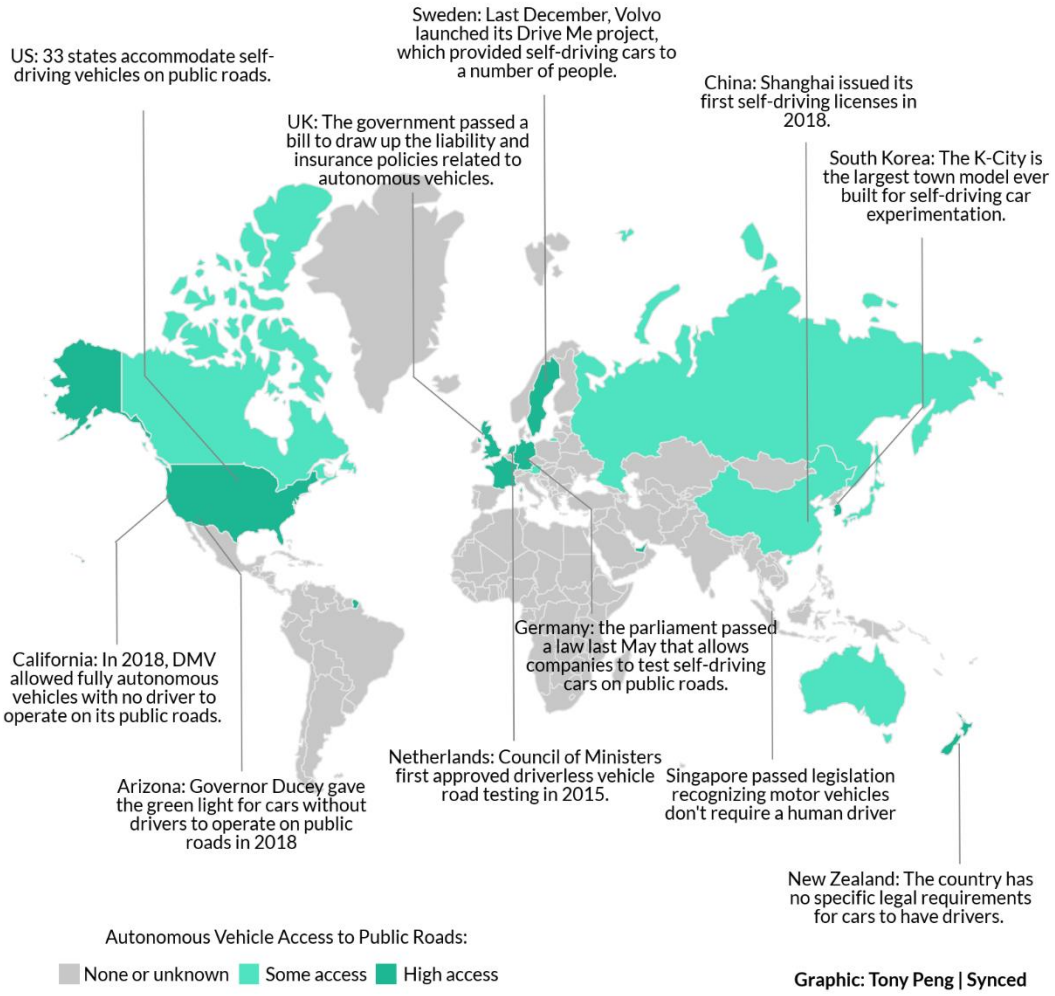


Ilustración 2.5: Gráfico mundial que representa las diferentes legislaciones de cada país [16]

Pasa seguir legalizando los vehículos autónomos, así como su desarrollo, el siguiente paso sería crear marcos reguladores que abarquen responsabilidades, tanto civiles como penales. Esto es debido a que, en caso de accidente, no se contempla aún responsabilidad [17]. Además, dentro de la implementación del código, aún no se ha impuesto nada que lo regularice. Es decir, en caso de atropello inminente, el vehículo debería tomar una decisión sobre a quién salvar, si al peatón que cruza o por el contrario a los pasajeros [18]. Por el momento esto lo decide de manera secreta cada compañía. Pero aquí de nuevo, aparece la ética, como ya se mencionó, es una de las partes más complicadas de abordar.

2.5. Entorno socioeconómico

En la actualidad está muy presente la idea de un futuro donde los vehículos autónomos son fieles protagonistas, y se puede afirmar que uno de los datos que con más relevancia se quiere conocer es el día que estos formarán parte de la sociedad.

A raíz de ello, se barajan diferentes ideas sobre con qué grado serán aceptados socialmente, y cuánto será el tiempo necesario para que dicho grado de porcentaje sea completo y satisfactorio.

Numeras empresas se atreven a decir una fecha de lanzamiento de los vehículos autónomos. En España, los datos más positivos señalan el año 2020, aunque es posible que esta fecha, por motivos de legalidad, se retrase en nuestro país.

Según las Naciones Unidad, cada año se alcanza un valor de 1,25 millones de muertes provocadas por accidentes de tráfico, y más de 50 millones de heridos mundialmente. La introducción en el mercado de este tipo de vehículos reduciría de manera considerable la cantidad mencionada [19]. Además, otro de los factores sociales más preocupantes es la cantidad de contaminación de un vehículo común, la cual se vería ampliamente disminuida debido a que se reduciría la búsqueda de aparcamiento en ciudad.

Respecto a la tecnología presentada en este proyecto, se espera una reacción social positiva. Aunque actualmente no se han dado situaciones de comunicación entre peatón y vehículo autónomo, se prevé que esta tecnología sea útil y obtenga buenos resultados, pudiendo asegurar una circulación en buenas condiciones con los humanos presentes en el entorno.

El impacto económico será sustancial en el área de infraestructuras creadas para los vehículos autónomos. Siendo positivos, lo que se espera de este tipo de vehículos es que sean capaces de convivir con vehículos de índole común. El futuro más cercano apunta que las infraestructuras deberán aumentarse debido a la incorporación de los vehículos autónomos, e incluso existe la posibilidad de crear vías donde únicamente circulen este tipo de vehículos.

Respecto a los núcleos urbanos, la interfaz de comunicación desarrollada intenta evitar posibles accidentes siendo los peatones capaces de conocer antes de cruzar cuándo hacerlo. Sabemos que evitando accidentes en la vía pública reduciremos sustancialmente

los costes directos e indirectos que supone la intervención a las personas heridas o colapso en la vía pública, entre otros factores.

También es importante no olvidar un vehículo autónomo está gestionado por un código desarrollado de manera privada por cada compañía. Esto da a entender que cada vehículo autónomo dispondrá de un tiempo de reacción y de una única posible decisión ante un accidente, por ello es importante que el peatón conozca sus posibilidades de cruzar en ese preciso momento.

Los datos arrojan que, en vías urbanas, durante el año 2017 en España, se han producido 1.067 accidentes mortales, falleciendo una cantidad de 1.200 personas y 4.837 heridos hospitalizados [20].

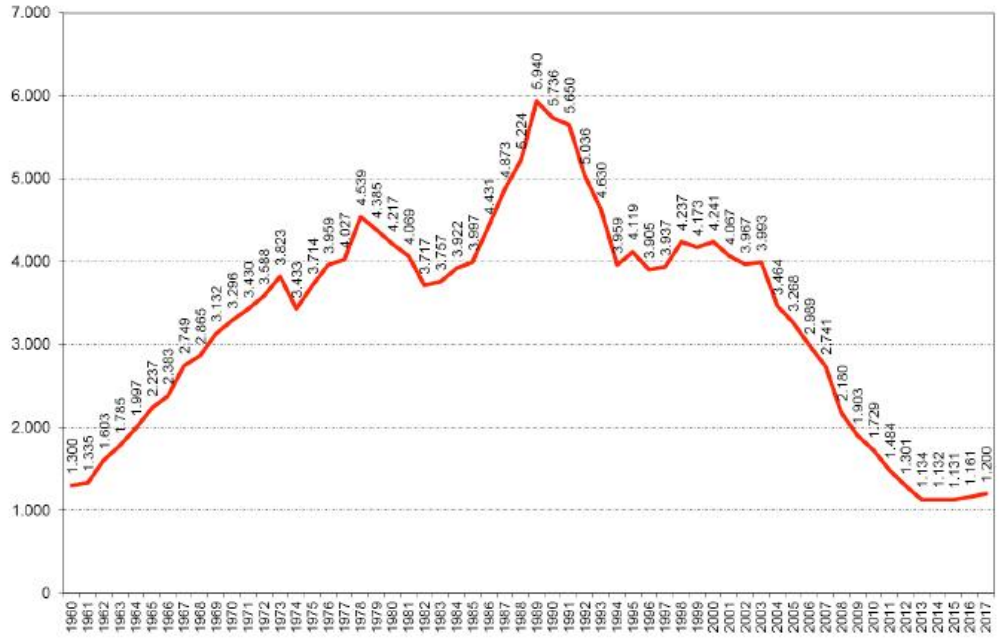


Ilustración 2.6: Fallecidos en vías urbanas en España [20]

3. DESCRIPCIÓN DE LA ARQUITECTURA UTILIZADA

3.1. Introducción

En este apartado de la memoria se describirán cada uno de los elementos que intervienen en el funcionamiento del proyecto.

En primer lugar, se hará un breve repaso por la arquitectura básica del vehículo autónomo utilizado, explicando cada una las partes esenciales en las que se basa su funcionamiento.

Posteriormente, se mostrarán los elementos software que son indispensables para el desarrollo del proyecto, y dentro de cada uno de ellos se explicará qué parte es útil y por qué se ha decidido su utilización.

3.2. Vehículo autónomo utilizado: iCab 1

Dentro del laboratorio de Sistemas Inteligentes de la Universidad Carlos III de Madrid nos podemos encontrar con dos tipos de vehículos en desarrollo, que serían, los vehículos terrestres autónomos, denominados iCab, y el vehículo inteligente basado en información visual, IvvI 2.0.

El vehículo autónomo iCab se trata de un carrito de golf modelo E-Z-GO, el cual ha sido modificado para convertirlo en un carrito de golf capaz de operar autónomamente [21]. Se encuentran a disposición de la universidad dos vehículos como este, iCab 1 e iCab 2.

Ambos carritos son idénticos, y a los que se les han hecho una serie de modificaciones, tanto mecánicas como eléctricas. Como, por ejemplo, la retirada del volante, el cual es dispensable dentro de los vehículos autónomos, para sustituirlo por un motor encoder que es capaz de controlar de manera electrónica la dirección del carrito. También fue retirado el pedal acelerador y sustituido por un sistema capaz de controlar la tracción, a través de un microcontrolador PIC.

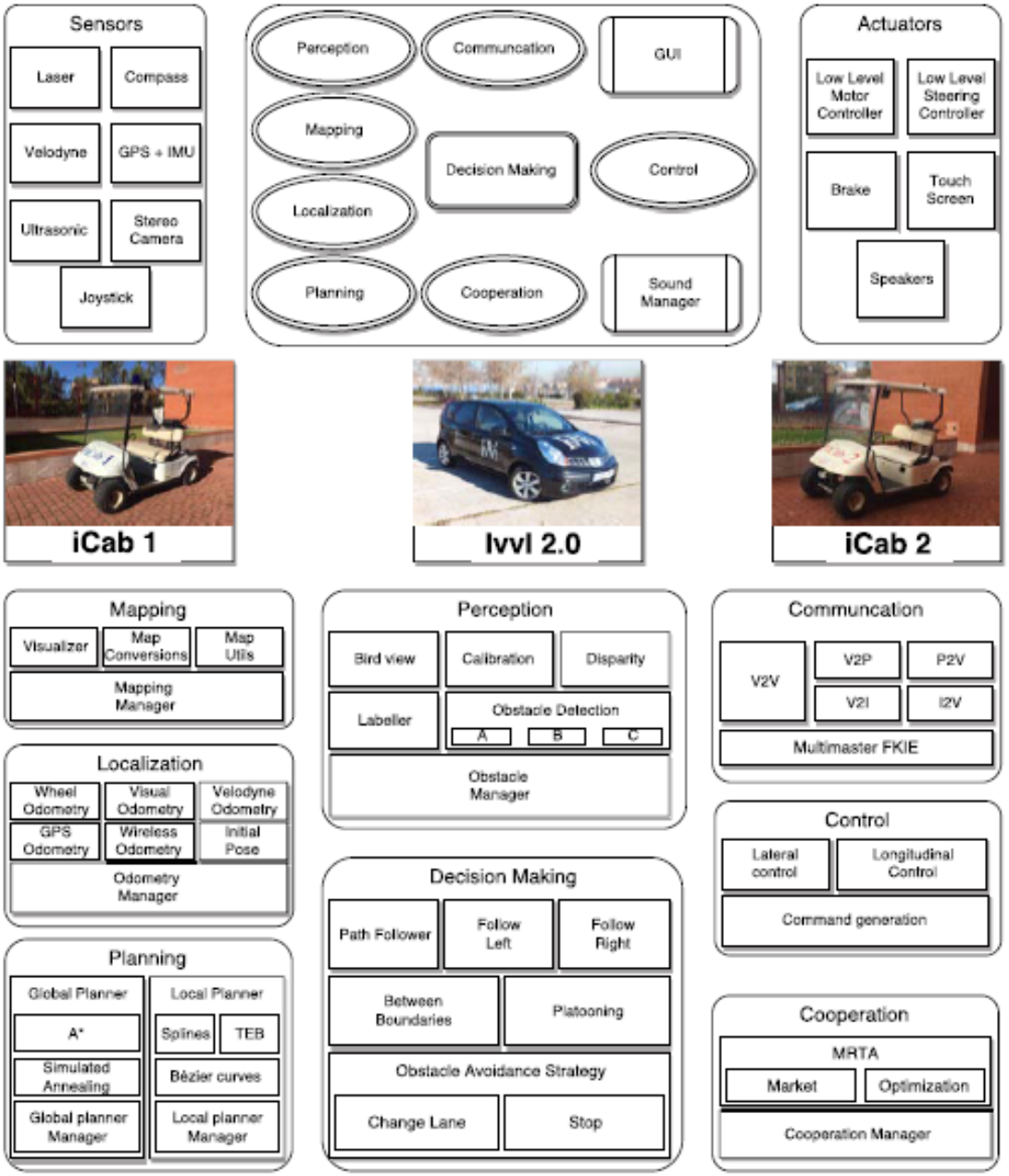


Ilustración 3.1: Arquitectura de los vehículos que se encuentran en desarrollo en la UC3M [22]

En la anterior imagen se muestra de forma generalizada cada una de las arquitecturas de los diferentes vehículos que se encuentran en el Laboratorio de Sistemas Inteligentes.

El vehículo autónomo utilizado en este proyecto es el iCab 1, y a continuación se describirán los elementos principales de los que está formado.

Los sensores utilizados en esta plataforma son [22]:

- Cámara estéreo configurable: se encuentra situada en la parte frontal superior del vehículo. En este caso, la cámara utilizada es la Bumblebee 2. Las principales ventajas que otorga son tanto el tamaño como la velocidad de procesamiento de las imágenes.

- Laser LIDAR de un solo plano: se encuentra situado en la parte frontal superior del vehículo. El láser LIDAR elegido es el modelo VLP-16, el cual es capaz de trabajar dentro de un rango de 360 grados, en 16 capas diferentes.

- Telémetro láser: en la parte frontal inferior se encuentra colocado este tipo de láser, utilizado para medir la distancia hasta un objeto que se encuentre delante. El láser incorporado en el iCab 1 es el modelo LMS 291.

- GPS, IMU y brújula: es importante colocar estos elementos en el vehículo autónomo para mantenerlos localizados, como se describió anteriormente. Todos ellos se encuentran en la parte superior del iCab 1.

- Sensores ultrasónicos: se encuentran colocados en la parte posterior del vehículo. Son utilizados para controlar, ya que envían y reciben el tiempo de vuelo, así como la distancia. El elemento utilizado para ello es modelo Pixhawk.

Además, debido a que no dispone de volante, este tipo de vehículos tiene incorporados también un Joystick. Es utilizado para controlar el vehículo cuando este se encuentre en modo manual.

En la siguiente imagen se representa dónde se encuentran colocados cada uno de los elementos hardware nombrados anteriormente.

Ilustración 3.2: Elementos hardware que conforman en vehículo autónomo iCab 1 [22]

Como se puede observar en la imagen, falta por describir dos elementos a los que aún no se hizo referencia: AAEON i7 y SKULL CANYON. Estos elementos forman también parte el hardware del vehículo y son los dos ordenadores que tiene integrado el iCab 1, indispensables para su funcionamiento.

La función de estos ordenadores es la de recibir información por parte de los sensores mencionados, y por medio de un software desarrollado, enviar la decisión a los actuadores.

Ambos ordenadores están integrados con el sistema operativo Ubuntu 16.04 y ROS con la versión Kinetic. Posteriormente se explicarán con más detalle ya que son indispensables en el desarrollo y funcionamiento de la interfaz propuesta.

Por último, cabe mencionar que ambos iCab tienen colocados en el interior del vehículo un botón de emergencia, que se trata de un botón electrónico con conexión Wifi.

3.3. Arquitectura del software utilizada

3.3.1. Ubuntu Linux

Como sistema operativo utilizado para el total desarrollo del proyecto se eligió Ubuntu 16.04, el cual es un sistema operativo de código abierto, es decir, cuya licencia es libre.

Está basada en Debian GNU/Linux, pero Ubuntu se caracteriza por su facilidad de uso, ya que se encuentra enfocado al uso de un usuario común.

La versión utilizada es 16.04, y aunque Ubuntu es un software que se encuentra en continuo de desarrollo, se eligió esta versión por motivos de compatibilidad con el vehículo autónomo iCab 1. Los ordenadores integrados en los carritos de golf autónomos tienen instalados esta versión, además de la mayoría los ordenadores de desarrollo de código que se encuentran en la universidad.

Para trabajar en Ubuntu siempre se hace uso de la terminal o consola, que se encuentra en cualquier GNU/Linux. La terminal es la encargada de interpretar órdenes y es el pilar básico de nuestro sistema operativo.

El formato básico de una orden en Ubuntu sería:

```
$ comando [-opciones] [argumentos]
```

La terminal es la herramienta principal que se usa a lo largo del desarrollo de la implementación del proyecto. En la siguiente tabla aparece un resumen de los comandos GNU/Linux más utilizados para navegar por el sistema de ficheros [23]:

Tabla 3.1: COMANDOS GNU/LINUX

ls	Se encarga de listar los archivos y directorios
ls -a	Enumera todos los archivos y directorios
mkdir	Crea un directorio
cd directorio	Se mueve hacia ese directorio
cd	Se mueve hacia el directorio home
pwd	Muestra la ruta del directorio actual
grep 'keyword' file	Busca dentro de un archivo la palabra clave que deseas
nano nombre_archivo	Abre un editor de texto dentro de la consola
comando --help	Mensaje de ayuda
clear	Limpia todos los mensajes de la consola

Para terminar, es importante conocer el uso del comando "sudo", *super-user-do*, el cual fue usado para la instalación de las herramientas necesarias para el desarrollo. Por seguridad, no todos los usuarios tienen acceso a la instalación de software en cualquier lugar del sistema de ficheros, que por defecto se encuentra restringido al usuario no administrador. Para obtener ese tipo de privilegios existe dicho comando, que otorga al usuario con el que se instala Ubuntu ser uno de los administradores del sistema.

Este comando es el que se utilizará más adelante para la instalación de ROS.

3.3.2. Robot Operating System

Para el desarrollo del software que utiliza el vehículo autónomo se ha utilizado el framework ROS, acrónimo inglés de *Robot Operating System*. El uso de ROS facilita la creación de código ya que proporciona al usuario el uso de bibliotecas y herramientas necesarias para el desarrollo de este, permitiendo así una perfecta comunicación entre la interfaz de control y el robot, que en nuestro caso sería el carrito de golf autónomo.

ROS también es una plataforma de código abierto, por lo que cualquiera que lo desee puede contribuir al desarrollo generando código de forma pública, y así mejorando y ayudando a otros usuarios [24]. Este es uno de los motores básicos en el desarrollo tecnológico, ya que favorece al rápido avance además de animar a otros usuarios a compartir sus proyectos.

Dentro de la web oficial de ROS se pueden encontrar un repositorio público donde poder descargar proyectos cedidos que sirven de ayuda al resto de usuarios en el desarrollo de sus propios proyectos; además de tutoriales, noticias y documentos básicos para la iniciación en ROS. Así es como ROS promueve la reutilización de código, tanto para personas para las que la robótica es una afición, a gente que se dedica profesionalmente a ello en numerosos campos de investigación.

ROS ofrece a los usuarios diferentes versiones, en este proyecto se utilizará la versión ROS Kinetic Kame (2016). La elección de esta versión es por motivos de compatibilidad, ya que los proyectos anteriores realizados y ya implementados en el vehículo autónomo, así como la versión de la que dispone este, es la ya mencionada ROS Kinetic Kame. Además, esta versión fue lanzada principalmente para su uso en Ubuntu 16.04, que como

se mencionó antes, es la versión del sistema operativo utilizada en el desarrollo del proyecto.

Al igual que se explicaron antes los comandos propios de Linux y su uso en la consola, ROS también dispone de sus propios comandos, los cuales facilitan su navegación por el sistema:

Tabla 3.2: COMANDOS ROS [25]

roscd directorio	Se mueve hacia ese directorio de ros
roscore	Ejecuta el sistema completo de ROS: permite la comunicación entre los nodos y siempre debe estar ejecutándose para el correcto funcionamiento del sistema
rosnode info nodo	Nos aporta información sobre ese nodo
rosnode kill nodo	Mata el proceso de ese nodo
rosnode list	Muestra los nodos que están ejecutándose
rosrun	Ejecuta el nodo
rostopic find	Encuentra el topic
rostopic list	Imprime información de los topics que están activos
rostopic type	Imprime el tipo de información del topic
roswtf	Nos permite conocer si algo está fallando

Los comandos citados permiten extender las funcionalidades de la terminal de GNU/Linux, es decir, no son excluyentes y ambos se pueden y se deben usar para navegar por las diferentes carpetas dentro del sistema operativo.

A continuación, se muestran los pasos que fueron seguidos para la creación y desarrollo del paquete que se instaló en el vehículo autónomo.

Lo primero que se debe de crear cuando se quiere empezar un proyecto en ROS es un espacio de trabajo, el cual contendrá los paquetes con nuestros archivos de código [26].

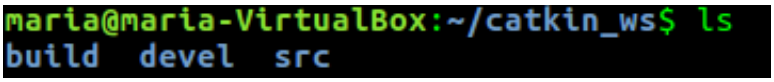


Ilustración 3.3: Archivos dentro del espacio de trabajo ROS

La imagen anterior muestra nuestro espacio de trabajo, comúnmente llamado "catkin_ws", y numerosos archivos en el interior. Los más importante a destacar son tres de las carpetas que aparecen, que serían [27]:

- Build: como su propio nombre indica, esta carpeta está relacionada con la compilación del propio proyecto. Desde esta carpeta se realizaría una llamada al "cmake" y "make" para configurar y crear los paquetes.

- Devel: esta carpeta recibe ese nombre como abreviación de la palabra en inglés development, la cual hace referencia a desarrollo. En esta carpeta se encuentran los programas ya compilados o, dicho de otra forma, todos los ejecutables y las bibliotecas necesarias del proyecto.
- Source: hace referencia a la carpeta que tiene como nombre src, dentro de la cual se almacenaran los paquetes de los proyectos y un archivo con el nombre CMakeList.txt.

Una vez que nuestro espacio de trabajo está creado, el siguiente paso es incluir dentro de él un paquete de trabajo, que irá dentro de la carpeta "src" como se muestra en la siguiente imagen:

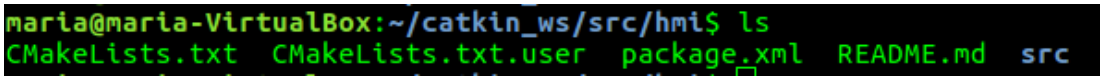


Ilustración 3.4: Archivos dentro del paquete "hmi"

El archivo que se encuentra dentro de nuestro paquete, denominado CMakeList.txt, es el que describe al sistema cómo se debe compilar el código y dónde se tiene que instalar. Es la entrada al sistema de compilación CMake cuando se crean paquetes.

Todo archivo CMakeList.txt está formado por la misma estructura, y es totalmente obligatorio seguirla o por el contrario los paquetes no compilarán correctamente. Cuando se crea un paquete, este archivo aparece automáticamente en su formato más básico. A medida que se va creando código, es imprescindible rellenar el archivo con los elementos necesario. Si nuestro programa utiliza como lenguaje C++, el archivo debe incluir los siguientes elementos [28]:

- Versión del CMake requerida.
- Nombre del paquete.
- Búsqueda de otros paquetes catkin que sean necesarios para la compilación.
- Declaración de archivos ROS de mensajes y servicios.
- Invocación para generar realmente mensajes y servicios añadidos.
- Declarar paquete catkin con sus dependencias en el tiempo de ejecución.
- Añadir bibliotecas y ejecutables para compilar, así como las dependencias.

El nombre que recibe nuestro paquete es "hmi", como abreviación inglesa de human machine interaction, dentro del cual se encuentran los archivos del código programado.

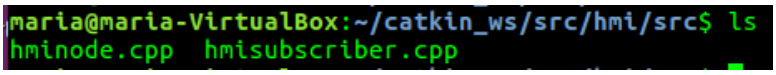


Ilustración 3.5: Archivos de código del paquete ROS

Por lo que finalmente, el espacio de trabajo con las subcarpetas creadas tendría la siguiente estructura:

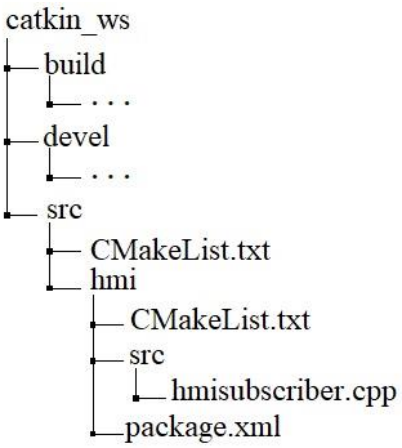


Ilustración 3.6: Estructura de carpetas del espacio de trabajo en ROS

El siguiente paso es comprender cómo conectar todos los elementos de nuestro sistema creando código por medio de ROS.

La forma de trabajo de ROS es la creación de una red donde es posible conectar todos los procesos del sistema. Esta red está formada por nodos, los cuales pueden interactuar entre ellos, ver la información que contienen y compartir datos [29].

Cada nodo se encarga de una función, por lo que es importante la creación de nodos independientes de tal manera que un sistema completo estará formado por numerosos nodos. Esto es preferible a tener un único nodo con muchas funcionalidades en el interior.

Todos los nodos se encuentran conectados en la red, para ello es importante la figura del "master". Es totalmente necesario ya que es el que permite a todos los nodos conectarse y mandarse mensajes, entre otras cosas.

Otro de los elementos importantes son los 'topics', los cuales son los elementos que permiten la conexión entre dos nodos o más. Un topic es simplemente un nombre que elige el desarrollador para identificar el tipo de mensaje que se quiere mandar entre los nodos.

La forma de trabajar es la siguiente: un nodo publica un topic, y otro nodo interesado en el mensaje que ofrece dicho nodo se puede suscribir a él usando el mismo topic. De esta forma aparece el nombre de nodos publicador y subscriptor [30].

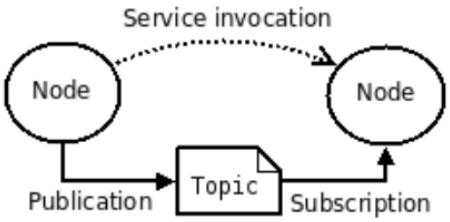


Ilustración 3.7: Esquema de comuniación entre nodos [29]

Desde la terminal, ROS ofrece un comando ("rqt_graph") que te muestra una gráfica de las conexiones en la red de nuestro sistema.

Ilustración 3.8: Ejemplo de conexión de nodos [31]

La imagen anterior es un ejemplo que aparece en la página oficial de tutoriales de ROS. En ella podemos ver dos nodos, marcados con los colores azul y verde, conectados por una flecha roja, que hace referencia al tópico. El nodo "/teleop_turtle" publica el tópico, y el nodo "/turtlesim" se subscribe a él.

Gracias a la conexión creada por ese topic, ambos nodos pueden compartir mensajes entre ellos.

Conociendo el funcionamiento básico de conexión, se crea el siguiente gráfico general del proyecto:

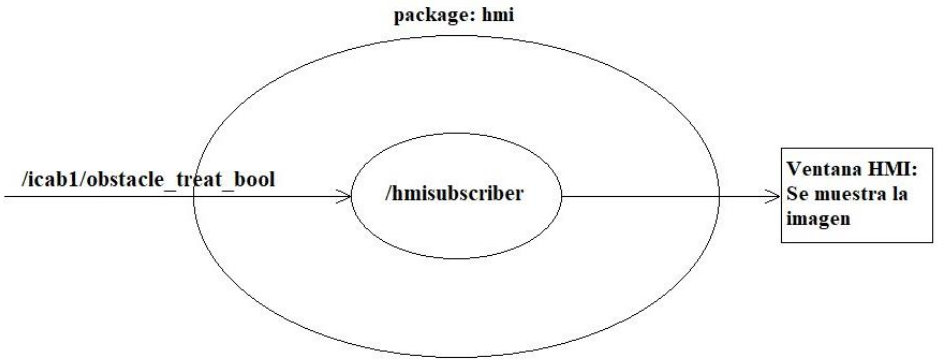


Ilustración 3.9: Representación del paquete desarrollado en ROS

La imagen muestra como dentro del paquete "hmi" se encuentra el nodo en funcionamiento "hmisubscriber". Este nodo es el encargado de realizar todas las funciones del proyecto. Está conectado por medio del tópico "/icab1/obstacle_treat_bool" a un nodo de otro paquete, el cual ya se encontraba instalado previamente en el vehículo autónomo.

Para finalizar y poner en funcionamiento la interfaz diseñada, dentro de la consola se debe poner en funcionamiento el máster, por medio del comando 'roscore'. Además, dentro del espacio de trabajo utilizado, se deben arrancar los nodos que necesitemos junto con el topic que les conecta.

3.3.3. Qt Creator

Como herramienta para desarrollar el código se hizo uno del programa Qt Creator, ya que se trata de un editor de C++, entre otros lenguajes, muy potente debido a sus características.

Qt Creator cuenta con multitud de ventajas, las cuales fueron decisivas a la hora de la elección de un editor para este proyecto

Se trata de un IDE de código abierto, por lo que cualquier persona que lo desee puede acceder a él de forma gratuita, aunque también tiene versión de pago con mayor cantidad de características incluidas. Además, como se ha ido mencionando a lo largo de la

memoria, todos los programas de código abierto tienen mayor desarrollo gracias al apoyo de sus usuarios y de las compañías que los sustentan.

La siguiente ventaja tiene que ver con lo visual, ya que, aunque la consola de Linux pone a disposición del usuario diferentes editores de texto, ninguno de ellos tiene las ventajas que aportan los IDE. Dentro de estas, se resaltan las siguientes [32]:

- Está formado por multitud de herramientas que permiten una rápida navegación a través del paquete, y sus diferentes archivos y elementos del código.
- Utiliza colores para diferenciar los diferentes elementos que conforman el código.
- Posibilidad de autocompletar palabras, en función de las librerías incluidas en el documento.
- Código estéticamente mejor diseñado de manera automática.
- En cuanto al depurado: capaz de ejecutar código línea a línea, mostrar dónde se encuentran los errores de programación o funcionamiento, así como poner puntos de interrupción de código donde el usuario desee, entre otras múltiples ventajas.

Y, para terminar, otra de las grandes ventajas por encima de otros editores, y por la cual se ha decidido utilizar este editor, es la de ser un entorno de desarrollo compatible con ROS. No se trata del único editor IDE compatible con ROS, también podemos encontrar otros como son el caso de Eclipse, CodeBlocks, Emacs, Vim, NetBeans, PyCharm, kDevelop y RoboWare Studio [33]. La elección de Qt Creator por encima de los editores nombrados anteriormente es porque se tenían conocimientos previos al desarrollo del proyecto en su utilización.

Incluyendo las librerías de ROS necesarias, Qt Creator es capaz de utilizar todas las ventajas nombradas anteriormente en códigos escritos en C++ usando ROS. Desde la propia consola de Linux puedes lanzar la interfaz de Qt Creator que usa ROS, para usar este editor más completo y facilitar la experiencia al usuario.

4. SISTEMA PROPUESTO

4.1. Fases de desarrollo

Para que cualquier proyecto se complete con éxito ha de pasar por tres etapas diferentes: una inicial, otra de desarrollo y una final.

Dentro de la etapa inicial se encuentra la asignación de un proyecto que se adecue a la persona que lo va a realizar. Con el proyecto asignado se empezaría a dar forma a los objetivos que necesita cumplir, y a cómo alcanzarlos.

Antes de afrontar esos objetivos se necesita una búsqueda de conocimiento, ya que el camino hasta llegar a ellos se trata de un camino también de aprendizaje. En este proyecto se necesitó una previa preparación con los elementos software utilizados, empezando por Linux.

Como sistema operativo elegido, Linux, se comenzó por mejorar el manejo, ya que se partía de un nivel básico de usuario. Por lo que, este se tuvo que incrementar empezando por tomar un primer contacto con la consola de Linux, hasta llegar a conseguir mejor agilidad en su uso como objetivo final.

Posterior a Linux, se empezó con el aprendizaje en ROS, del cual no se tenía ninguna base acerca de su utilización, por lo que el proceso fue más lento y complicado.

Después de una amplia documentación en Linux y ROS, ya que la base de conocimiento en Qt Creator era suficiente, se dio por finalizada la etapa inicial. Etapa que se puede resumir como la adquisición de objetivos y conocimientos para la realización de la siguiente fase, la fase de desarrollo.

En la fase de desarrollo se incluye desde la implementación del código hasta las posteriores pruebas, tanto en el ordenador como reales utilizando así el vehículo autónomo iCab 1. Las pruebas son una de las partes más importante de cada proyecto, ya que ayudan a decidir cuáles son los posibles cambios en el código que mejorarán la optimización de este.

La idea inicial de este proyecto estaba clara, pero había diferentes caminos hasta llegar al diseño final presentado. El primer diseño estaba formado por dos nodos, como se muestra en la siguiente imagen.

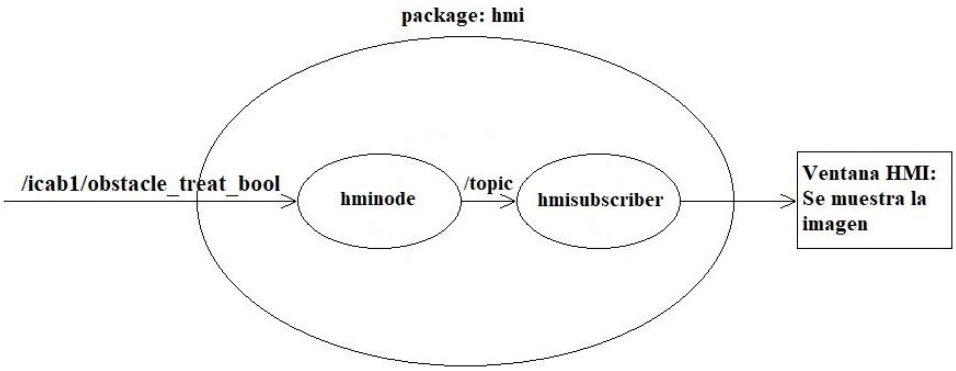


Ilustración 4.1: Representación inicial del paquete en ROS

El paquete estaba formado por dos nodos, el primero, "hminode" estaba subscrito al topic que publicaba otro paquete ya implementado en el coche, y este, a la vez, publicaba un topic al nodo "hmisibscriber" quien, en función de eso, mandaba una imagen u otra a la pantalla del vehículo autónomo.

Aunque aparentemente aportaba mayor claridad de entendimiento, después de probarlo con parte del equipo encargado en el desarrollo, se decidió cambiar por un paquete de un único nodo. Este fue el diseño final, donde el nodo se encarga de subscribirse al topic que publica otro paquete implementado en el coche, y en función de los mensajes recibidos, muestra la imagen adecuada para cada situación. Esta decisión mejoró la optimización del programa, ya que como se acaba de comentar, se pasó de tener dos nodos, a un único nodo, capaz de realizar todas las funciones.

Otra de las cosas que se mejoraron una vez probado fue la adaptación de la imagen a la pantalla. Inicialmente, se crearon imágenes con un tamaño estándar, pero al colocar la pantalla, estas adaptaban a la resolución, modificando así la imagen. Por ese motivo, se tuvieron que crear nuevas imágenes con el tamaño adecuado en función de la resolución de la pantalla, para evitar que las imágenes aparecieran distorsionadas y pixeladas.

También hubo pequeños retoques, como la decisión de a qué distancia de seguridad debería el coche dar la alarma de que un peatón se encontraba cerca de él, fueron también modificados. Para decidir esto, había que entender el funcionamiento del otro paquete ya instalado con el que se encuentra unido el creado en este proyecto, así como los mensajes que recibe y cómo los envía. Gracias a esto, el paquete creado recibiría un mensaje que haría tomar la decisión de mostrar una u otra imagen, aportando luz al peatón que se encontrase delante a partir de cierta distancia.

Las imágenes que se prueban no son las definitivas, ya que en esta etapa de desarrollo se haría un estudio sobre el diseño de las imágenes mostrar al peatón, ya que deben ser claras y rápidas de entender por cualquier tipo de usuario. Lo primero que hay que hacer antes de enfrentarse a un nuevo reto, es leer y documentarse acerca de ello. Múltiples compañías están trabajando en la idea que se desarrolla en este proyecto, por lo que existen multitud de variantes y soluciones propuestas.

Con todo el proyecto preparado se da por finalizada esta etapa y se da pie a la etapa final. En esta fase se realizan múltiples pruebas reales, dentro de la universidad y con personas ajenas al proyecto. La finalidad de estas pruebas es la de aportar una opinión nueva, y saber si este nuevo avance ayuda a los peatones y les aporta confianza el uso de vehículos autónomos, o por el contrario crea mayor confusión. Para ello se realizaron una serie de cuestionarios posteriores a la interacción de las personas seleccionadas, y se analizaron para saber cuáles eran los resultados obtenidos.

Por último, la realización de esta memoria, la cual resume todo el trabajo realizado en los últimos meses.

4.2. Diseño de las imágenes a mostrar

Para el diseño de las imágenes de las pruebas reales se tuvieron en cuentan los diferentes escenarios que se quieren analizar.

Por parte del equipo de desarrollo se plantearon tres escenarios diferentes. Una primera prueba sin la interfaz del proyecto presente en funcionamiento, es decir, el vehículo autónomo iCab 1 no muestra ninguna imagen a los peatones. Con esto se pretende comparar la situación previa al proyecto, y obtener unos resultados que nos permitan conocer si realmente es importante la comunicación entre vehículo y peatón que se plantea.

La segunda y tercera prueba sí que tienen implementada la interfaz, mostrando una imagen al peatón cuando este es detectado. La diferencia entre ambas pruebas es la imagen que se muestra.

La segunda prueba está basada en los colores rojo y verde. La pantalla mostrará el color verde cuando el peatón ha sido detectado, y rojo, cuando no. De esta prueba se espera bastante incertidumbre por parte del peatón ya que, aunque instintivamente las

personas asociamos el color verde a bien/correcto y rojo a mal/error, es complicado que un peatón que tiene un primer contacto con este tipo de mensajes sea capaz de asociarlo a cruzar o no cruzar la vía.

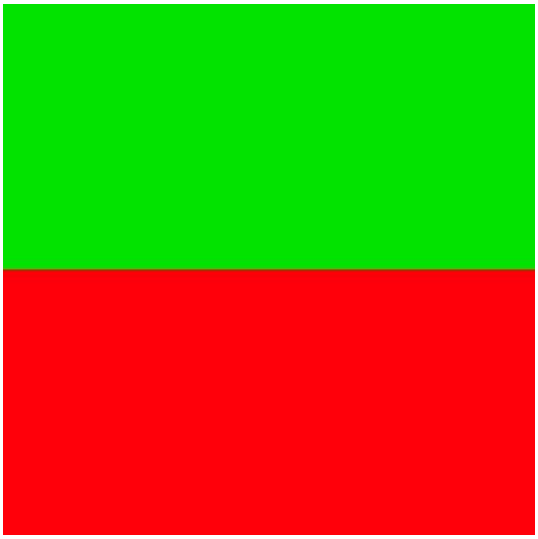


Ilustración 4.2: Imagen experimento verde/rojo

Para la tercera y última prueba se elaboraron dos imágenes asociadas al contacto visual. Se han realizado experimentos por investigadores del sector previos al proyecto que certifican que el contacto visual entre peatón y conductor es un punto importante para que el peatón tome la decisión de cruzar la carretera en un paso de peatones no regulado por semáforos.

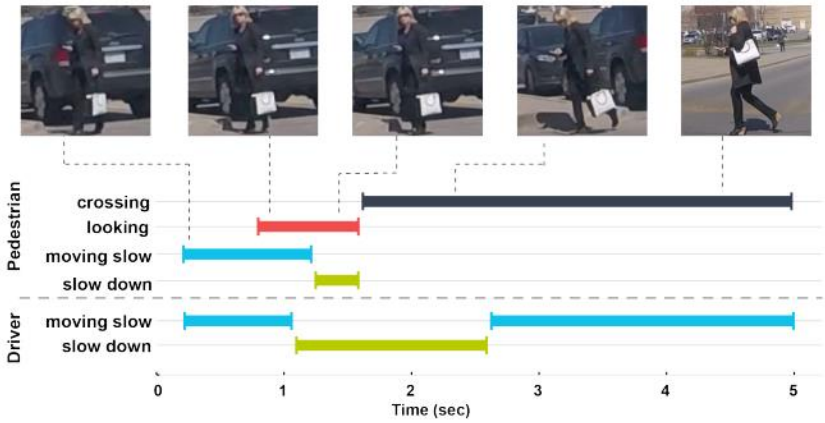


Ilustración 4.3: Comportamiento peatón y conductor en un cruce [34]

La imagen anterior es el resultado temporal medio que se obtuvo en uno de los experimentos mencionados. Para su realización se contó con la presencia de más de 650 peatones en un periodo de conducción de aproximadamente 240 horas. Como se puede observar, la imagen está dividida en las acciones realizadas por parte del peatón y las del

conductor. Analizando la línea de tiempo, inicialmente el conductor avanza despacio, pero ralentiza su velocidad cuando se encuentra con el peatón. De igual manera, el peatón ralentiza su velocidad y gira su cabeza para comprobar si es seguro cruzar, como la parte roja de la gráfica indica, y así poder continuar su camino en caso de ser afirmativo [34].

De ahí es de donde surge la idea de crear unas imágenes que simulen el contacto visual con el conductor en la ausencia de este. Se diseñaron dos imágenes, una con los ojos abiertos, que sería mostrada cuando el iCab detectase al peatón, y la otra con los ojos cerrados para mostrar en la situación inversa.

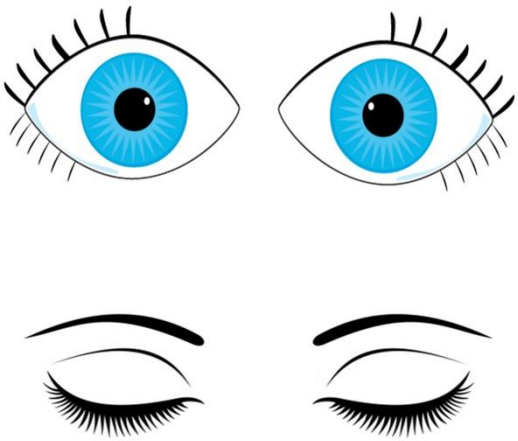


Ilustración 4.4: Imagen experimento ojos abiertos/cerrados

Durante la realización del proyecto, el fabricante de automóviles Jaguar Land Rover sacó a la luz su propuesta para crear más confianza en el ser humano cuando se encuentre frente a un vehículo autónomo sin conductor. Dicha propuesta se basa en un vehículo autónomo al que se le han incorporado ojos que son capaces de seguir con la mirada al peatón que se aproxima [35].

Aunque no es la única propuesta que ha sido lanzada a la luz por parte de fabricantes y desarrolladores de vehículos autónomos. Hoy en día se puede ver muy presente la importancia y necesidad de crear interfaces que sean capaces de comunicarse con los peatones. Tanto es así, que se piensa que los vehículos autónomos podrán ser usados por la población cuando esta confíe cien por cien en ellos y para eso, es necesario que el resto de los usuarios externos al vehículo autónomo también lo hagan.

Ilustración 4.5: Interfaz de interacción creada por Land Rover [36]

Por último, para concluir el diseño de las imágenes, surgió la idea de complementar ambas imágenes mostradas en una única imagen. Aunque la imagen resultante no se planteó ponerla a prueba, si se incluyó dentro de una de las preguntas de los cuestionarios realizados para entregar al público participante del experimento.

Ilustración 4.6: Imagen complementaria de las realizadas para el experimento

Durante la prueba de imagen y pantalla que se realizó días previos al experimento, aparte de conocer la resolución y tamaño ideal que debería tener cada una de las imágenes para no deformarse, se probó la visibilidad de estas en la nave cubierta de la universidad.

Es importante conocer de antemano cómo iban a ver las imágenes los participantes del experimento, llegando a la conclusión de que una imagen complementaria de ambas ideas iba a aportar mayor claridad en el entendimiento. De tal forma, se decidió que sería buena idea incluir la imagen en los cuestionarios para así conocer qué opinión al respecto tendrían los participantes.

4.3. Funcionamiento del paquete

En este apartado se va a hacer una explicación más extensa y clara sobre el funcionamiento del paquete, dándole más importancia al código programado.

Para empezar, se explicará el diagrama de estados del código, el cual nos da información acerca de los pasos que sigue este cada vez que se ejecuta el programa, y los diferentes estados por los que pasa, de manera teórica. Posteriormente se detallarán las librerías incluidas en el código para aclarar la funcionalidad de cada una de ellas y por qué son necesarias. Para terminar, se repasarán algunas partes importantes del código, incidiendo más en aquellas que hacen referencia el entorno de trabajo, ROS.

4.3.1. Diagrama de estados

La siguiente imagen explica de manera generalizada cada uno de los estados en los que nuestro programa se encuentra en cada momento.

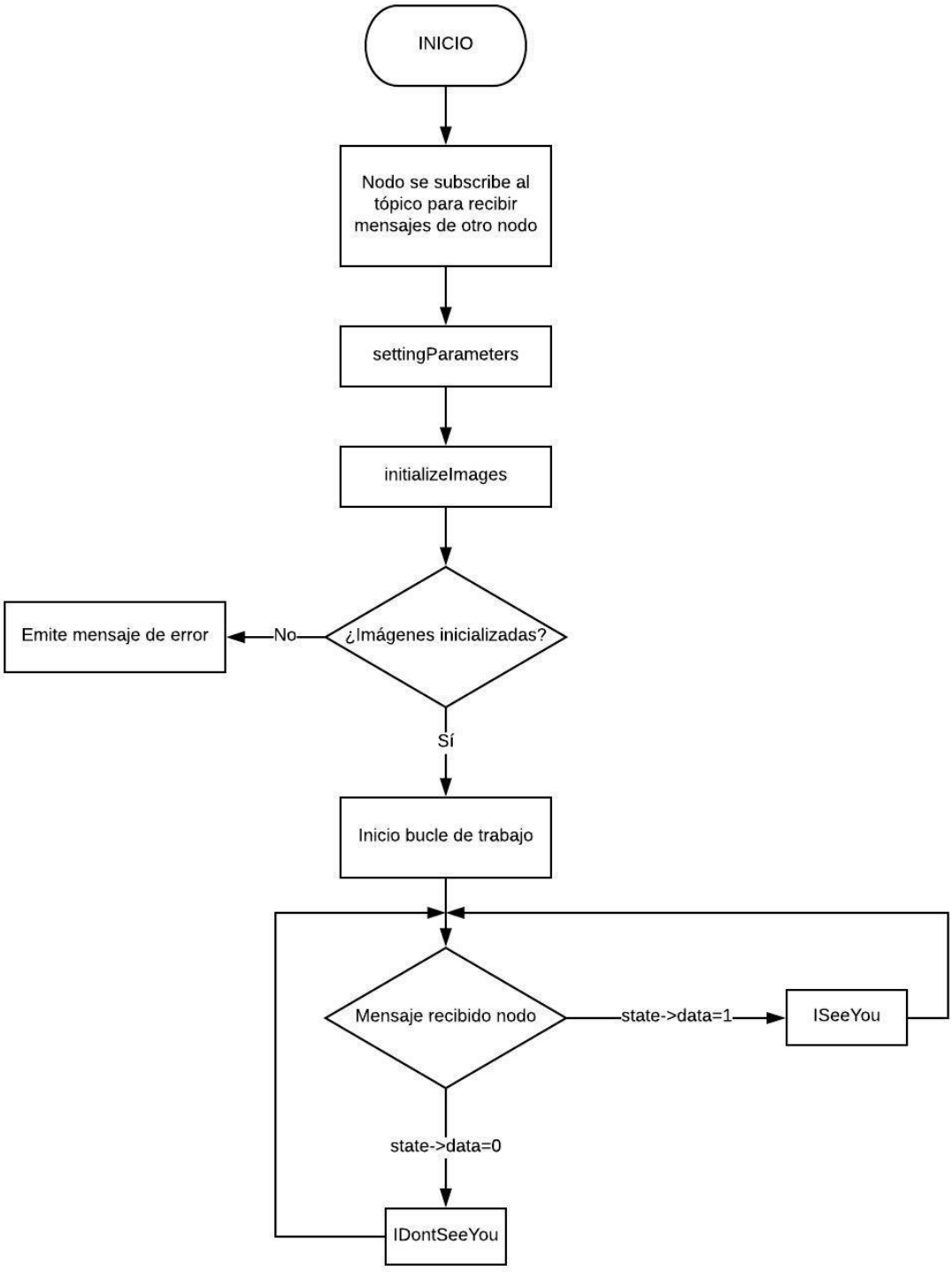


Ilustración 4.7: Diagrama de estados del sistema desarrollado

El programa parte de un estado inicial, de reposo. Se pone en funcionamiento cuando se le invoca desde la terminal usando los pasos explicados con anterioridad, pasando así al siguiente estado.

Como indica la imagen, el primer estado por el que pasa nuestro algoritmo hace referencia a la conexión con otro nodo ya implementado en el vehículo autónomo. Para ello, se conecta al topic, que nosotros previamente hemos estudiado y creado, para recibir mensajes por parte de otro nodo dentro de otro paquete. Es importante que este sea el primer paso, ya que los mensajes recibidos a partir de ese topic son los que influirán en la solución esperada por el programa.

Una vez conectado, ambos nodos provenientes de distintos paquetes se mantendrán enviando y recibiendo mensajes respectivamente hasta el final de la ejecución del programa.

Después se pasa al siguiente estado, que consiste en incluir en el código las imágenes que posteriormente se mostrarán por la pantalla al peatón. Inicialmente los archivos de las imágenes se encontraban en el interior del paquete “hmi” pero, aunque esto no daba problemas en el funcionamiento, dentro de un paquete de ROS únicamente deben estar los archivos de código necesarios, con sus diferentes carpetas necesarias y aquellas creadas automáticamente (véase punto 3.3.2).

Debido a que las imágenes no tienen cabida ahí, se crea un estado en el que, usando parametrización en ROS, se adquieren esas imágenes que se encuentran instaladas en otra carpeta dentro del software del vehículo.

Una vez recogidas, se introducen dentro de una matriz previamente creada en el código, y se pasa al siguiente estado, la inicialización. Las imágenes se muestran gracias a Open CV, biblioteca libre utilizada en visión artificial. La importancia de este estado recae en la necesidad de saber si realmente se están tomando de forma correcta esas imágenes.

En el diagrama de estados se crea un estado de decisión, simbolizado por el rombo. Este estado bifurca el programa en dos caminos. Si la imagen fue inicializada correctamente el programa seguirá ejecutándose. En caso contrario, se mostrará un mensaje de error, que detendría el programa, ya que, si las imágenes no son capaces de inicializarse, el estado final no podría mostrar la imagen adecuada.

Con todo inicializado y funcionando se da comienzo al bucle de trabajo. Donde, en función de los mensajes recibidos por el otro nodo, se toma una decisión. El nodo anteriormente mencionado publica un mensaje de tipo booleano, es decir, publica 0 o 1 en función del código programado en el dicho nodo.

Lo que debemos conocer por parte de ese nodo es que cuando publica un mensaje positivo, o un 1, quiere decir que el vehículo ha detectado un peatón. Cuando publica un falso, o un 0, significa lo contrario, que el vehículo no ha detectado ningún peatón.

En función de eso, como podemos observar en el esquema, se publicará una imagen u otra y se retomaría el bucle. Eso indica que este programa es infinito, ya que no queremos que en ningún momento deje de recibir mensajes y publicar imágenes.

El programa únicamente dejará de funcionar cuando el controlador humano lo decida.

4.3.2. Librerías utilizadas

La cabecera del código es la que incluye las librerías necesarias para el correcto funcionamiento del programa.

Podemos definir una librería como un conjunto de algoritmos que han sido previamente creados y puestos a disposición del programados. Dichos algoritmos facilitan la realización de ciertas operaciones que se utilizan con asiduidad, evitando al programador tener que crearlas continuamente.

Estas se sitúan en las primeras líneas de código, seguidas de "#include", ya que es necesario incluirlas en la fase de preprocesado.

La cabecera de nuestro proyecto es la siguiente:

```
#include "ros/ros.h"
#include "std_msgs/Bool.h"
#include "std_msgs/String.h"
#include <string.h>
#include <opencv2/opencv.hpp>
```

Ilustración 4.8: Cabecera programa C++

Como se puede observar en la imagen, son necesarias cinco librerías diferentes para el correcto funcionamiento de nuestro proyecto.

- "ros/ros.h": librería que incluye todas las cabeceras necesarias para el uso de las instrucciones más comunes en ROS [37].

- "std_msgs/Bool.h" y "std_msgs/String.h": librerías utilizadas para enviar mensajes estándares en ROS [38]. En estas dos cabeceras estamos permitiendo el intercambio de mensajes de tipo booleanos y de cadena de caracteres.
- <string.h>: librería estándar de C++ que incluye multitud de funciones que permiten trabajar con cadenas de caracteres [39].
- <opencv2/opencv.hpp>: como se mencionó anteriormente, la librería utilizada que permite trabajar con imágenes es Open CV [40]. Para poder incluir las funciones necesarias para su uso es necesario incluir previamente esta cabecera en el programa.

4.3.3. Descripción del nodo

Para el funcionamiento de un nodo en ROS debe incluir una serie de elementos en el código. El objetivo de este apartado es explicar cada uno de ellos, para comprender cual es su utilidad y por qué son completamente necesarios.

Dentro del programa principal del nodo deben aparecer las siguientes instrucciones en el siguiente orden:

```
int main(int argc, char **argv)
{
  ros::init(argc, argv, "nombre_del_nodo");
  ros::NodeHandle n;
  ros::Subscriber inputdata =   n.subscribe("/topic", 1000,
chatterCallback);

  ros::Rate rate(10);

  while (ros::ok())
  {
    waitKey(1);
    ros::spinOnce();
    rate.sleep();
  }

  return 0;
}
```

Ilustración 4.9: Bucle principal ROS/C++

- "ros::init(argc, argv, "nombre_del_nodo")": es necesario incluir dentro del paréntesis el nombre con el que queramos inicializar el nodo, así como los argumentos. Al incluir "argc" y "argv" permitimos al programa realizar cualquier argumento de ROS [41].

- "ros::NodeHandle n": esta instrucción es necesaria para crear el principal punto de acceso a las comunicaciones del nodo con el sistema ROS, permitiendo interactuar con los topics de la red. En nuestro programa, nos permite la suscripción al topic proveniente de otro nodo. El nombre "n" es el identificativo que otorga el programador y debe ser único [42].

- "ros::Subscriber inputdata = n.subscribe("/topic", 1000, chatterCallback)": esta línea de código hace referencia que el nodo esta subscrito a un tópico. Es importante mencionar que un nodo puede estar subscrito a varios topics, así como publicar varios mensajes a través de otros topics. Es decir, puede ser subscribirse y publicar utilizando más de un topic. En este programa solo aparece esta línea de código ya que el nodo solo es un subscriptor de un único topic. Dentro del paréntesis lo primero que aparece es el nombre del topic, seguido de un número que indica la cantidad máxima de mensajes capaces de almacenar y procesar, y por último la llamada a la función "chatterCallback". La funcionalidad de esta función es la de realizar una acción cada vez que un mensaje es recibido. El tamaño de la cola de mensajes que se pueden almacenar es elegido por el programador. Cuando se llegue al máximo seleccionado se empezarían a eliminar mensajes más antiguos para empezar a incluir los nuevos que entren al nodo [43].

Las siguientes líneas que aparecen en el código hacen referencia a los ciclos o bucles, que indican la periodicidad con la que recibimos mensajes:

- "ros::Rate rate (10)": es la forma utilizada para especificar a qué frecuencia en Hercios se ejecuta el bucle "while" . En nuestro caso la frecuencia elegida fue de 10Hz [43].

- "while (ros::ok())": inicio del bucle, que no se detendrá hasta que ROS emita un falso, creando así un bucle infinito [43]. En nuestro programa, las formas de emitir un falso son externas al código y sería ingresando por teclado Ctrl+C. Aunque existen otras dos formas de crear un falso al programa, que se daría el caso cuando se introduce en la red otro nodo con el mismo nombre o llamando a "ros::shutdown()" desde otra parte del programa.

- "ros::spinOnce()": utilizado para bucles infinitos que tienen como función procesar algo. Es decir, incluir este tipo de bucle le permite a ROS procesar los mensajes que recibe. En nuestro programa se incluye "waitKey(1)", que se trata de un comando de openCV usado en C++. El número que se incluye entre paréntesis es el "retraso" en milisegundos y es necesario llamar a este comando para procesar el bucle de evento, que en nuestro programa sería la actualización de la imagen a mostrar [44].

- "rate.sleep()": comando que se asegura que se cumple el ciclo, ya que hace que el bucle duerma el tiempo restante necesario para alcanzar nuestra tasa de publicación fijada, en este caso, en 10Hz [44].

5. PRUEBAS Y RESULTADOS

5.1. Pruebas

El experimento consiste en poner a prueba la interfaz diseñada en el iCab 1 en el campus de la universidad Carlos III de Madrid, y posteriormente analizar los resultados obtenidos. Para ello se realizaron una serie de cuestionarios que se entregaron a cada uno de los participantes al finalizar cada una de las pruebas.

Inicialmente, para la ambientación, se planteó la idea de colocar en el exterior una lona a modo que simulara un paso de peatones. De esta forma, los peatones sabrían, sin interacción con las personas encargadas de la realización del experimento, que deberían pasar por ahí. Al carecer de ella, se decidió llevar a cabo la prueba sin ningún tipo de ambientación, ya que al ser una primera prueba lo realmente importante era conocer la reacción inicial de cada uno de los participantes. Esto nos serviría de ayuda para mejorar las posibles siguientes pruebas que se fueran a ejecutar en un futuro, y conocer los puntos débiles de las pruebas presentes en este proyecto.

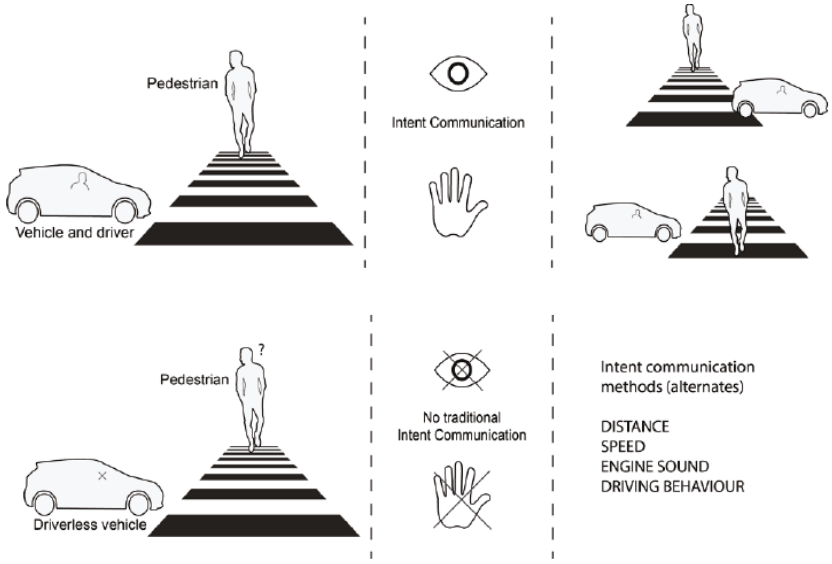


Ilustración 5.1: Comportamiento peatón ante un VA [45]

La idea en la que se basa el experimento es la que se plantea en la anterior imagen. Por parte del peatón, una comunicación ofrecida por el conductor del vehículo es importante para tomar la decisión de cruzar un paso de peatones. Pero, en caso de la inexistencia de conductor, ¿qué métodos ayudarían al peatón a tomar dicha decisión de cruzar la vía y cuál será su reacción? Las respuestas a la pregunta son los objetivos que se pretenden alcanzar con los experimentos.

Para la preparación de la pantalla, inicialmente en pruebas previas, se decidió la colocación de una pantalla cuyo tamaño era pequeño pero que no daba problemas para el entendimiento de las imágenes diseñadas.

Después de realizar pruebas exteriores con el equipo del experimento, se llegó a la conclusión de que había problemas de nitidez en las imágenes cuando el sol incidía directamente sobre ella. Esto provocaba que en determinadas situaciones no se apreciara claramente el mensaje que la pantalla mostraba.

Para solucionarlo, se decidió usar la pantalla incorporada actualmente en los vehículos iCab y colocarla en la parte delantera del vehículo. Al ser bastante más grande que la pantalla inicialmente escogida, se ofrecía mejor calidad en la nitidez de las imágenes, incluso en aquellas zonas donde el sol incidía frontalmente.

Como lugar para realizar las pruebas se escogió una de las calles principales que atraviesan el campus ya que, aparte de ser bastante frecuentada por estudiantes de la universidad, también es de uso público por los habitantes de la localidad de Leganés. Con ello, nos aseguramos de que no todos los participantes del experimento conocen el vehículo iCab, ni saben que se trata de un vehículo autónomo, pudiendo obtener reacciones reales de una primera interacción del peatón con un vehículo de este nivel.

A modo de simulación de paso de peatones, se escogieron unos baldosines con diferente color al resto, y se escogían participantes al azar que estuvieran interesados en realizar una de las pruebas propuestas. Cada participante realizaba una única prueba. Posteriormente, se les hacía entrega de un cuestionario que completaban en el mismo momento.

Ilustración 5.2: Imagen tomada del experimento realizado

Como se comentó anteriormente, se realizaron un total de tres pruebas diferentes y para cada una de ellas se contó con cinco participantes diferentes. Tanto los análisis que se van a realizar como las conclusiones que se van a obtener, están basados en las respuestas personales de cada uno de ellos.

5.2. Resultados y análisis de los formularios

A continuación, se presentarán los resultados obtenidos por parte de los participantes del experimento. Se analizarán cada una de las preguntas de los cuestionarios con una gráfica indicativa, una breve explicación y finalmente, una conclusión general de las pruebas.

5.2.1. Prueba 1: Sistema sin interfaz

Para la primera prueba realizada no se tuvo en cuenta la interfaz diseñada en el proyecto. Por lo tanto, la idea principal por la que se realiza dicha prueba es la necesidad de conocer el comportamiento humano ante un vehículo sin conductor y así poder compararlo posteriormente con el comportamiento usando la interfaz diseñada.

Para ello, como se comentó previamente, el peatón tendrá que cruzar la calle elegida dentro de la universidad, mientras el vehículo autónomo iCab 1 se aproxima. Posteriormente, se le entregó a cada uno de los participantes un cuestionario del cual se obtuvieron los siguientes resultados en función de cada una de las preguntas.

1. ¿Habías visto o conocías el vehículo autónomo iCab antes del experimento?

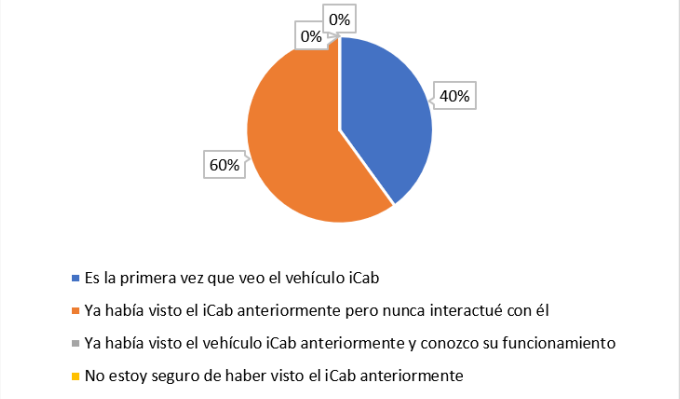


Ilustración 5.3: Resultados sobre si se conocía el iCab previamente prueba 1

Es importante esta pregunta, porque necesitamos saber de antemano si los participantes reconocían en vehículo autónomo, o si habían formado parte de algún otro experimento realizado.

El objetivo del experimento es el de poder realizar una interacción con los participantes ajenos al proyecto, y como se aprecia en la gráfica, ninguno de ellos conocía el funcionamiento del vehículo.

2. ¿Te diste cuenta de que el vehículo iCab se trata de un vehículo autónomo?

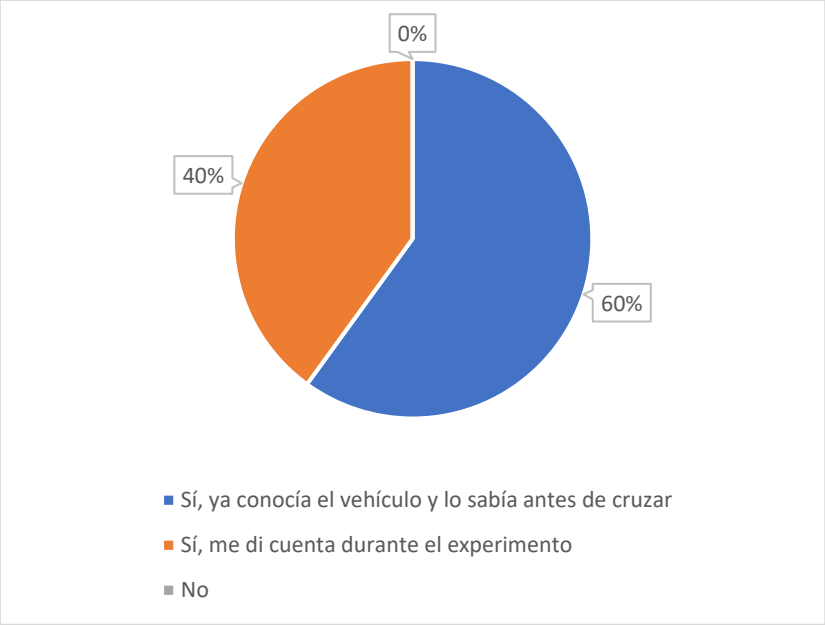


Ilustración 5.4: Opinión de los participantes sobre el conductor prueba 1

Esta pregunta fue incluida en el cuestionario con el objetivo de conocer cuántos participantes realmente se fijaban en el conductor del vehículo antes de cruzar, e indirectamente, en las características del vehículo. Una vez más, es importante conocer este tipo de opiniones debido a que la interfaz que se desarrolla está orientada en este camino, es decir, la sustitución de la reacción de un conductor y la repercusión en el peatón.

3. ¿Cruzaste la vía cuando el vehículo estaba desacelerando?

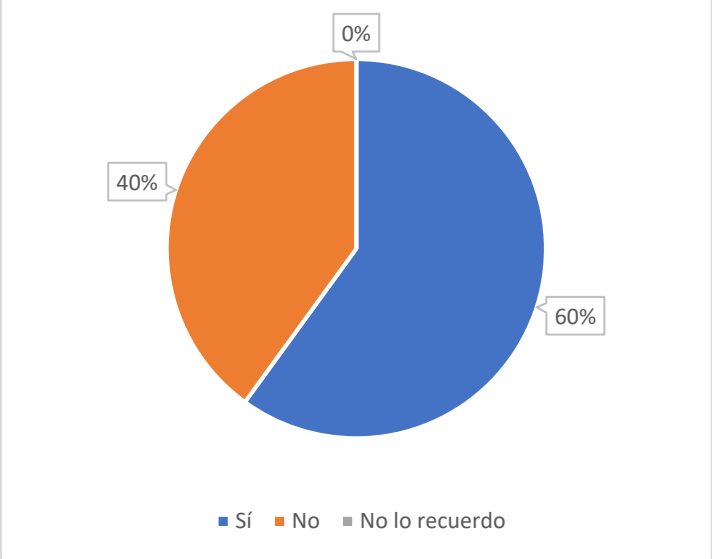


Ilustración 5.5: Resultados sobre cuándo empezaron a cruzar los participantes prueba 1

4. ¿Esperaste a que el vehículo estuviera completamente parado para cruzar?

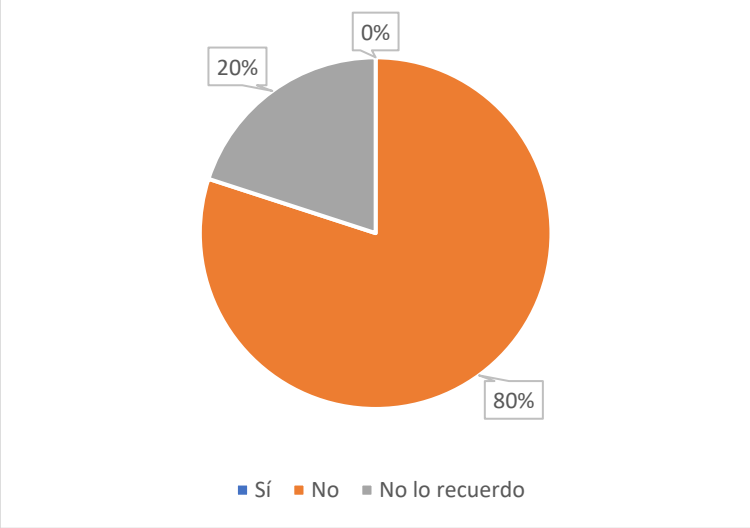


Ilustración 5.6: Resultados sobre la reacción de los participantes prueba 1

Las anteriores dos preguntas están completamente ligadas, y de ambas se obtuvieron resultados contaminados. Aunque en un principio se espera esa parte de duda por parte del peatón, según estas dos preguntas, se puede observar que fueron bastante decisivos a la hora de cruzar, es decir, no se esperaron a la total detención del vehículo.

Esto se debe a un problema del ángulo de detección inferior del láser que se encuentra incluido en el vehículo. La distancia programada para tomar a un peatón como detectado

es ajena al proyecto, ya que se encuentra dentro de otro de los paquetes implementados, pero no es la correcta. El sistema considera al peatón como detectado cuando se encuentra frente al vehículo. Realmente eso no es lo que buscamos, ya que lo que se quiere lograr en este proyecto es dar seguridad antes de cruzar, es decir, cuando el peatón se encuentra esperando para hacerlo.

5. Es una escala del 1 al 5, ¿tuviste dudas a la hora de cruzar?

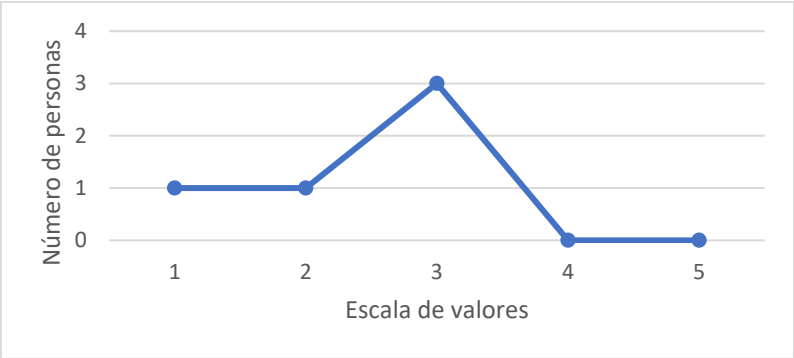


Ilustración 5.7: Resultados sobre dudas a la hora de cruzar prueba 1

Donde 1 hace referencia a nada de dudas, y 5 a completamente falta de confianza, se puede sacar como conclusión que la mayor parte de los participantes tuvo ciertas dudas a la hora de cruzar.

Aunque esta respuesta puede arrojar cierta parte de error por lo comentado respecto a las dos preguntas anteriores se sabe, mediante la realización de otras pruebas previas ajenas al proyecto, que por lo general los peatones muestran cierta desconfianza. De hecho, por esos motivos se decidió realizar la investigación y desarrollo de la interfaz de este proyecto.

6. Si tuviste dudas, ¿puedes seleccionar los motivos?

En esta pregunta, se les ofreció a los participantes una serie de posibles motivos. Entre ellos, los más seleccionados hacían referencia a la falta de conocimientos acerca de si realmente habían sido detectados. Por ese motivo, gran parte de ellos también opinó que realmente no estaban seguros de si el vehículo iba a frenar a tiempo.

Esto confirma que realmente es necesaria la creación de una interfaz que sea capaz de comunicar el vehículo sin conductor con el peatón, proporcionando así la falta de seguridad que reflejan en las respuestas del cuestionario.

7. Basándote en tu experiencia, si el vehículo pudiera hacerte saber si fuiste detectado, ¿te sentirías más seguro a la hora de cruzar?

El objetivo de esta pregunta, aparte de conocer la opinión al respecto, es dar a conocer públicamente a personas ajenas en la investigación de sistemas de conducción autónoma, que en la realidad se están valorando ideas al respecto.

Basado en mi experiencia, es importante que los peatones sean conscientes de la existencia de esta tecnología para un correcto funcionamiento.

5.2.2. Prueba 2 y 3: Sistema con interfaz

En ambas pruebas, el vehículo mostraba una de las imágenes diseñadas, o fondo verde y rojo, u ojos abiertos o cerrados, respectivamente. Los resultados de ambas pruebas se presentan en conjunto, excepto aquellas preguntas que hacen referencia a las imágenes mostradas.

Estas pruebas se encuentran divididas en dos partes, la primera parte está asociada al comportamiento habitual del peatón en un cruce no señalizado con semáforos, y la segunda parte, se quiere conocer la reacción del peatón ante el experimento con el iCab.

Primera parte: Comportamiento habitual diario

8. En una escala del 1 al 5, ¿mantiene contacto visual con los conductores de los vehículos antes de cruzar la carretera en un paso de peatones (del tipo sin semáforos)?

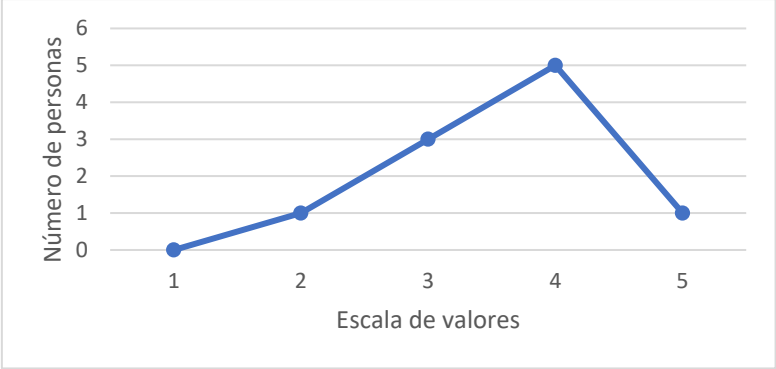


Ilustración 5.8: Resultados sobre el contacto visual habitual al conductor

Donde 1 hace referencia a nada de contacto visual, y 5 a contacto visual siempre que se cruza la carretera, se puede sacar como conclusión que la mayoría de los experimentados tiene contacto visual diario con los conductores, aunque no siempre.

9. En su día a día, ¿suele esperar a que el vehículo esté completamente parado para cruzar?

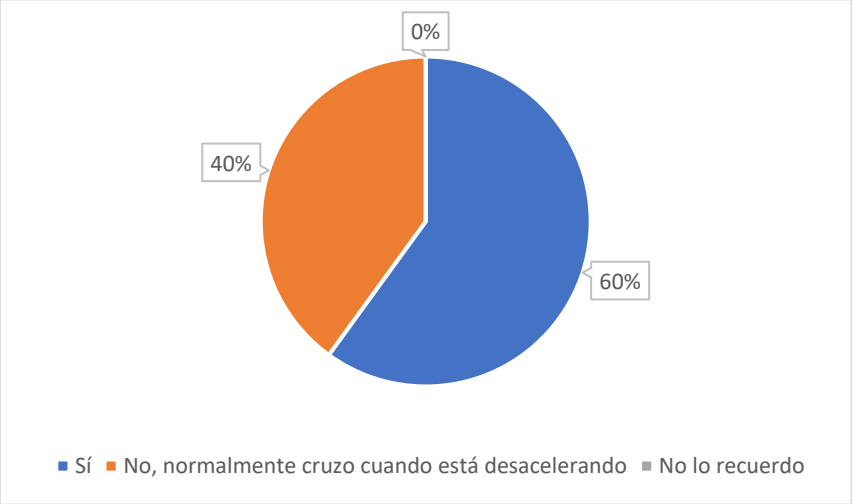


Ilustración 5.9: Resultados sobre cuándo suele cruzar el peatón

Estos resultados nos indican que normalmente, antes de cruzar la vía, la mayoría de los participantes del experimento suele esperar a que el vehículo se encuentre totalmente parado. El 40% de ellos, suele empezar a cruzar cuando el vehículo empieza a perder velocidad.

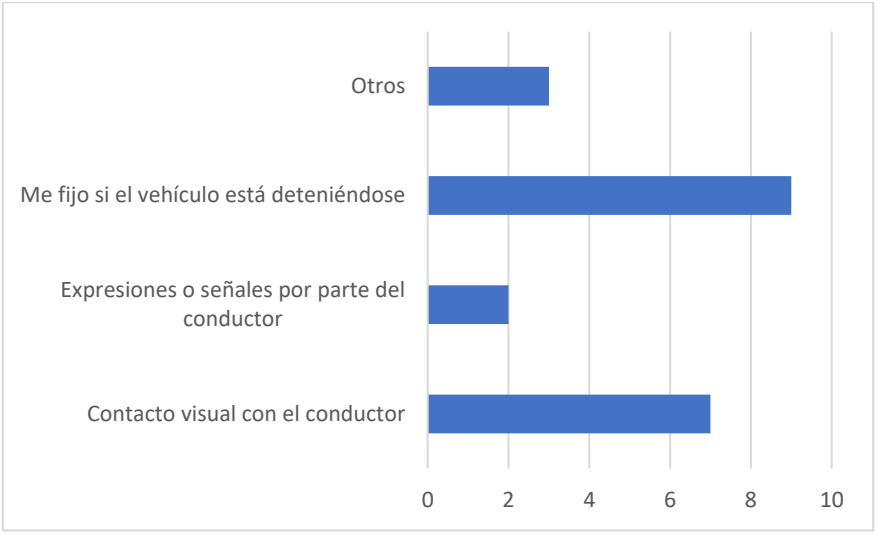


Ilustración 5.10: Resultado sobre qué motivos son importantes para cruzar la vía

Se les entrevistó también acerca de los motivos que les hacían tomar la decisión de cruzar en su día a día. La mayoría de los participantes indicó que para ellos era importante tener contacto visual con el conductor, ya que eso implicaba haber sido vistos. Aunque la

mayor parte de ellos también indicaba que necesitaban ver que el vehículo estaba perdiendo velocidad.

Dentro del apartado otros, se mencionó la importancia del comportamiento de otros vehículos y peatones de alrededor, así como una serie de obstáculos que obligasen al conductor a frenar.

La importancia de este breve cuestionario radica en conocer si realmente es importante la figura del conductor sobre el peatón en el momento de tomar la decisión sobre si cruzar, o mejor esperar, en un paso de peatones del tipo sin semáforos.

Se puede sacar como conclusión lo que muchos estudios previos ya obtuvieron [34]: realmente si es importante que el peatón reciba una idea sobre si ha sido o no reconocido, tanto por el conductor, como en nuestro proyecto, por el vehículo autónomo.

Segunda parte: Comportamiento durante el experimento

1. ¿Habías visto o conocías el vehículo autónomo iCab antes del experimento?

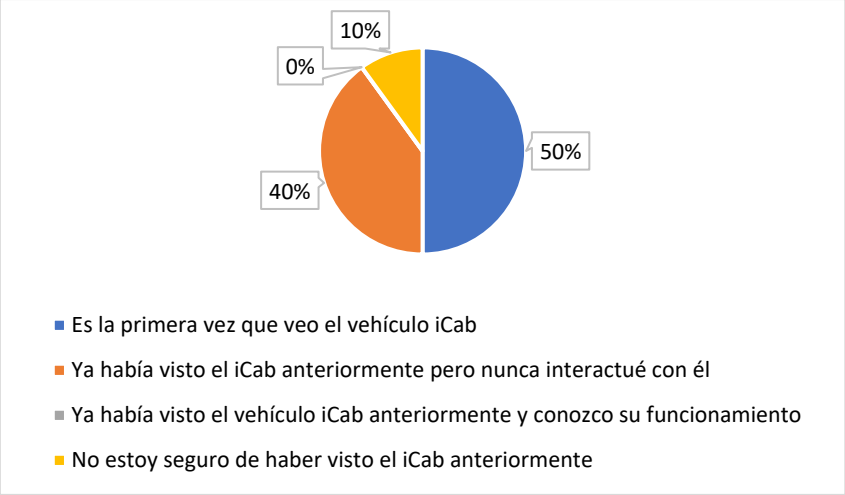


Ilustración 5.11: Resultados sobre si se conocía el iCab previamente prueba 2-3

Aunque como se puede apreciar un gran porcentaje de personas había visto previamente el iCab, confirman en el cuestionario que en ningún momento interactuaron con él. La mitad de los participantes nunca habían coincidido con el iCab.

Los valores que arroja esta pregunta son satisfactorios ya que, aunque es complicado no haber visto previamente el vehículo en el recinto de la universidad se demuestra que ninguno de los participantes conocía nada acerca de cómo iba a reaccionar el vehículo ante su presencia.

2. Antes de cruzar, ¿te fijaste si había un conductor dentro del vehículo?

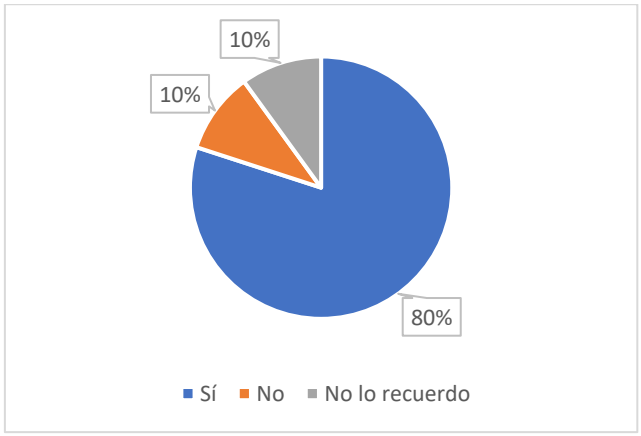


Ilustración 5.12: Opinión de los participantes sobre el conductor prueba 2-3

La gran mayoría de los participantes afirman que sí se fijaron en ello. Estos resultados no son nada sorprendentes ya que, durante la prueba, uno de los miembros del equipo se introduzco dentro del vehículo por motivos de seguridad.

A parte de que, al no existir paso de peatón, la selección de los voluntarios fue realizada también por los miembros del equipo, quiénes les indicaban dónde y cuándo deberían ponerse a cruzar.

Como conclusión, los resultados obtenidos no son concluyentes, ya que todos los participantes eran conscientes de que se les estaba realizando una prueba.

3. ¿Te diste cuenta de que el vehículo iCab se trata de un vehículo autónomo?

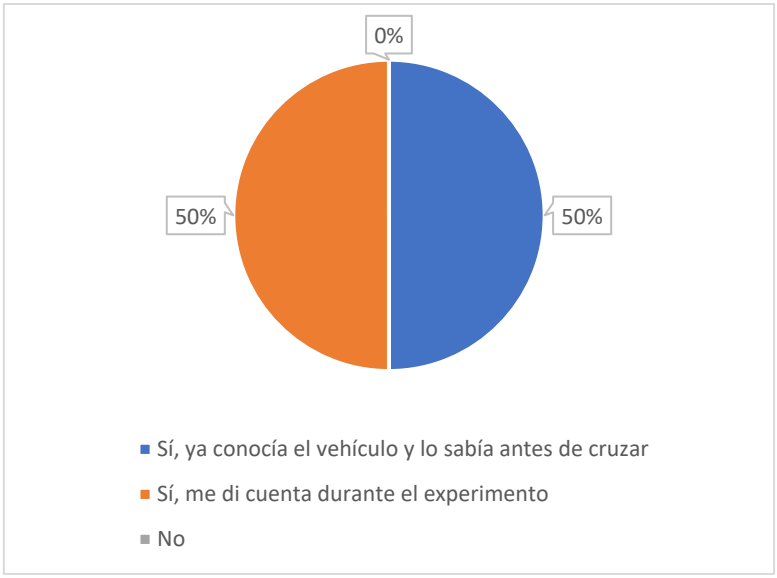


Ilustración 5.13: Resultados sobre si conocían que el iCab se trata de un vehículo autónomo prueba 2-3

A pesar de encontrarse un miembro del equipo dentro del vehículo, la mitad de los participantes se dieron cuenta de que se trataba de un vehículo autónomo, ya que carece de volante, entre otras cosas apreciables a simple vista. La otra mitad ya conocía que se trataba de un vehículo autónomo en desarrollo.

Ninguno de los participantes dudó sobre si se trataba de un vehículo autónomo o no.

4. ¿Cruzaste la vía cuando el vehículo estaba desacelerando?

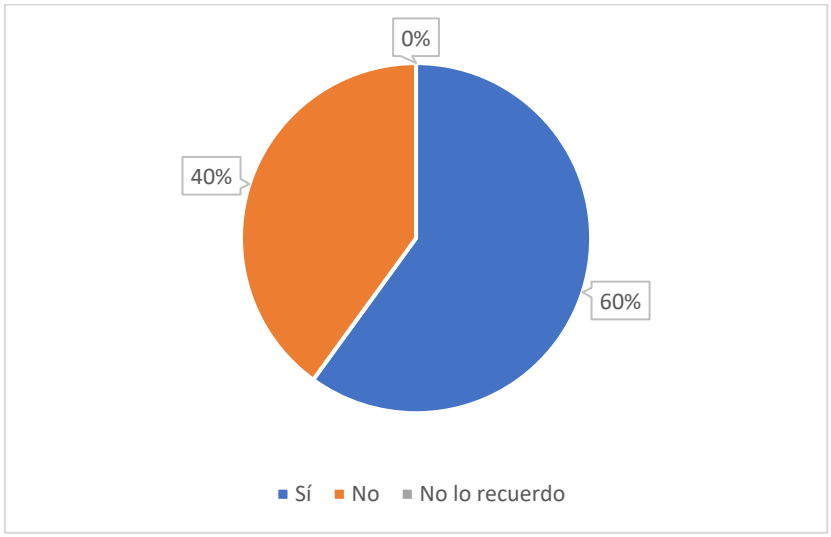


Ilustración 5.14: Resultados sobre cuándo empezaron a cruzar los participantes prueba 2-3

5. ¿Esperaste a que el vehículo estuviera completamente parado para cruzar?

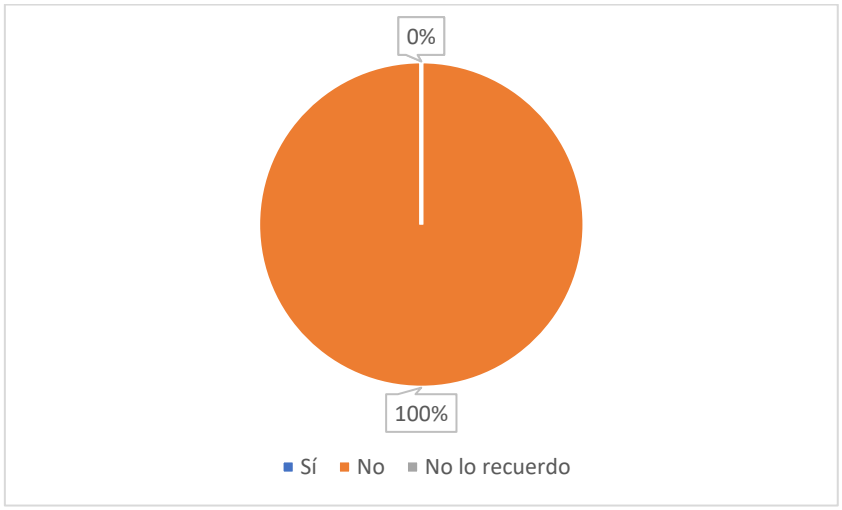


Ilustración 5.15: Resultados sobre la reacción de los participantes prueba 2-3

Como se comentó en la primera parte del experimento, estas dos preguntas arrojan resultados contaminados porque, aunque aparentemente se puede dar a entender que los peatones cruzaron con total confianza, en realidad eso no es cierto. Los miembros del equipo éramos quiénes les ofrecíamos esa confianza animándolos a cruzar.

6. Es una escala del 1 al 5, ¿tuviste dudas a la hora de cruzar?

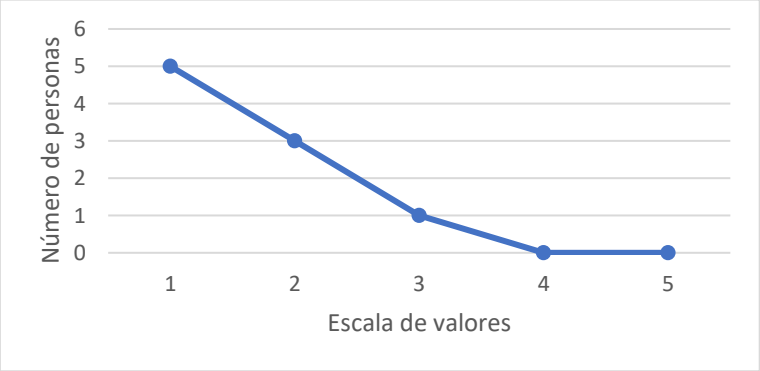


Ilustración 5.16: Resultados sobre dudas a la hora de cruzar prueba 2-3

Donde 1 hace referencia a nada de dudas, y 5 a completamente falta de confianza, se puede sacar como conclusión que la mayoría de los experimentados no tuvieron nada de dudas y un 40% de ellos sí tuvo ciertas dudas.

Esta respuesta también arroja algo de error, por todo lo comentado anteriormente, incluyendo también que uno de los miembros del equipo debía encontrase dentro del vehículo por motivos de seguridad.

7. ¿Viste la imagen que muestra el monitor del vehículo autónomo para indicar que has sido detectado?

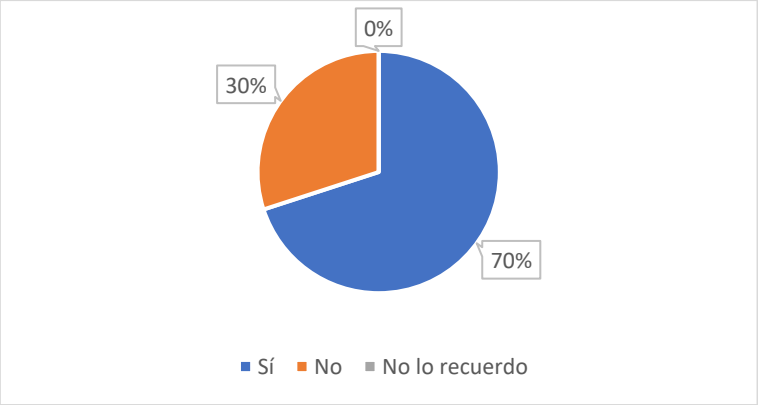


Ilustración 5.17: Resultados sobre si fue visto el monitor del iCab

Aunque según los resultados la mayoría de los participantes sí que vio el monitor colocado para mostrar la imagen, una gran parte de ellos no se fijó.

Esto crea dudas sobre si sería necesario incluir algún elemento más que indique la detección del peatón, o si realmente, con el monitor es suficiente y los peatones no se fijaron en él debido a la excesiva confianza en que no iba a ocurrir ningún accidente por nuestra presencia.

8. Si contesto sí, ¿cree que la imagen que muestra el monitor es suficientemente clara?

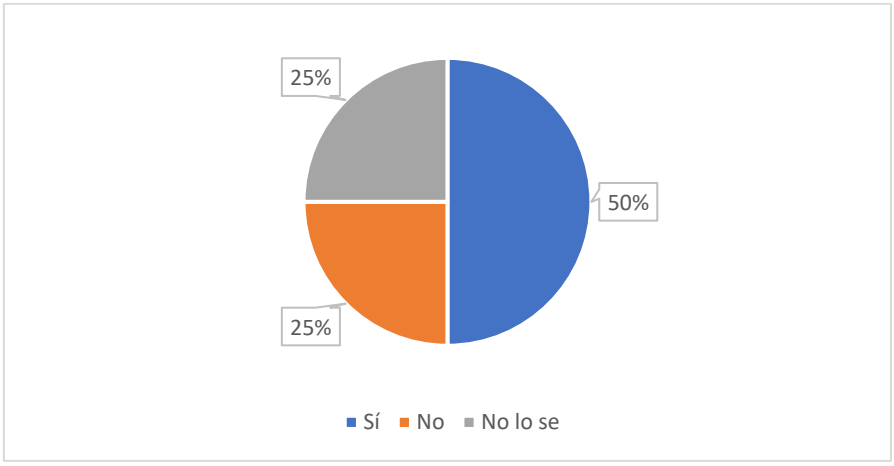


Ilustración 5.18: Resultados obtenidos para la imagen ojos abiertos/cerrados

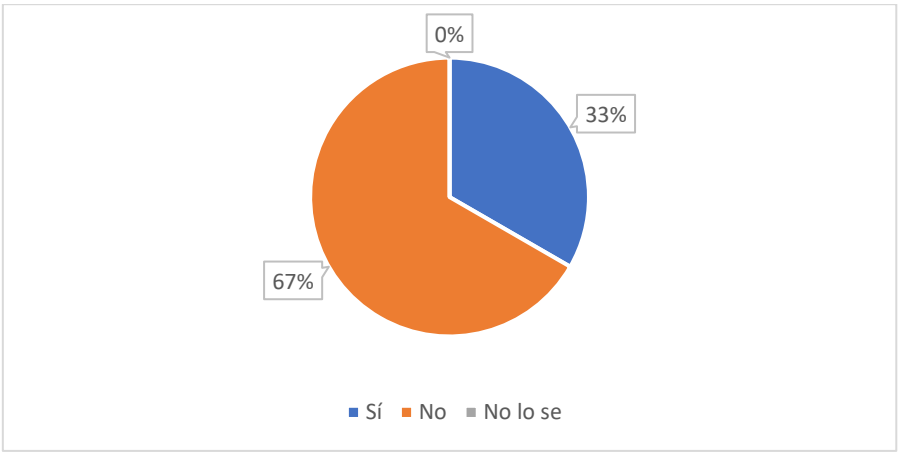


Ilustración 5.19: Resultados obtenidos para la imagen verde/rojo

Los resultados obtenidos son que para la mayoría de los participantes que vieron la imagen que mostraba la pantalla, es mucho más claro de entender aquella donde aparecen representados los ojos abiertos y cerrados, que representar la detección únicamente con imágenes verdes y rojas.

9. ¿Cree que mostrar una imagen realmente ayudaría al peatón a tener más confianza a la hora de cruzar la carretera cuando se aproxima un vehículo sin conductor?

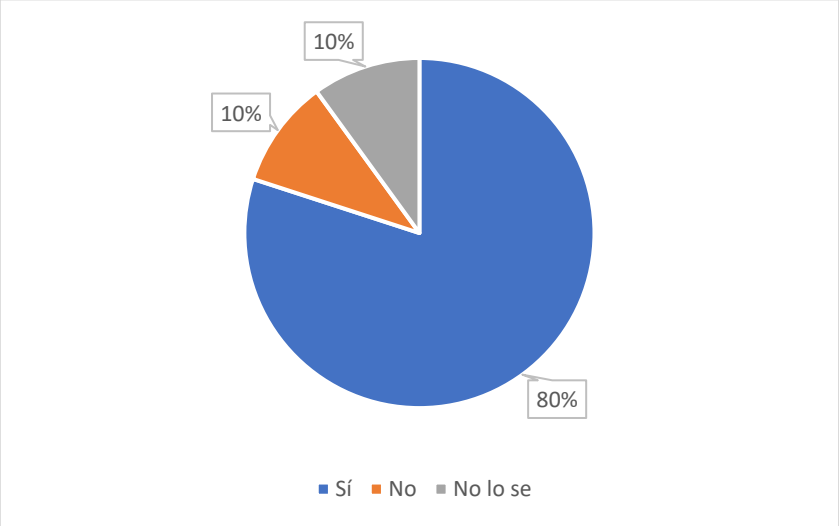


Ilustración 5.20: Resultados sobre la necesidad de incluir una imagen

Por lo general, se puede apreciar que la respuesta de los participantes es positiva y realmente consideran necesario la inclusión de algo informativo, en este caso una imagen, que les haga saber que fueron detectados por el vehículo autónomo.

10. A continuación, se muestran varias imágenes posibles que indican si el peatón ha sido detectado por el vehículo autónomo. Señale cuál es la que le resulta más fácil de entender.

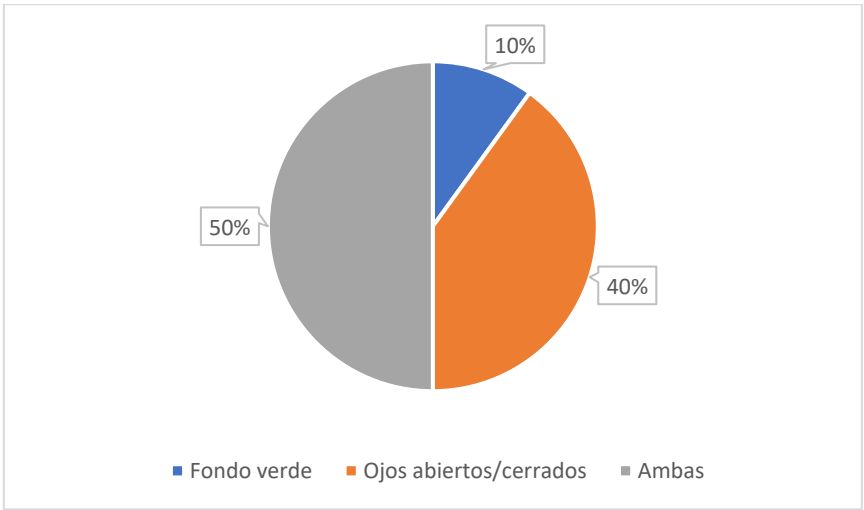


Ilustración 5.21: Resultados sobre qué imagen mostrar

Para finalizar, se mostró a cada uno de los participantes las tres imágenes diseñadas para conocer cuál sería la ideal y con mejor entendimiento para ellos. La imagen

resultante fue la que no se puso en práctica en el experimento, pero que será valorada para experimentos posteriores.

La conclusión general obtenida por ambas pruebas es que es totalmente necesario una correcta ambientación para su realización, ya que si ven presentes al equipo es muy posible que las dudas que puedan surgir desaparezcan, ya que son conscientes de que no permitiremos que ocurra ningún fallo.

Respecto a la interfaz, la respuesta que se obtuvo de los participantes fue muy positiva. Aunque en los cuestionarios se refleje cierto desconocimiento, se puede afirmar que este es debido a que es la primera vez que se encuentran en una situación similar. Como la mayoría de los avances en esta área no constan de patentes es difícil que, con un primer contacto de la interfaz con el peatón, sepan identificar que existe una pantalla que indica cuando son o no detectados.

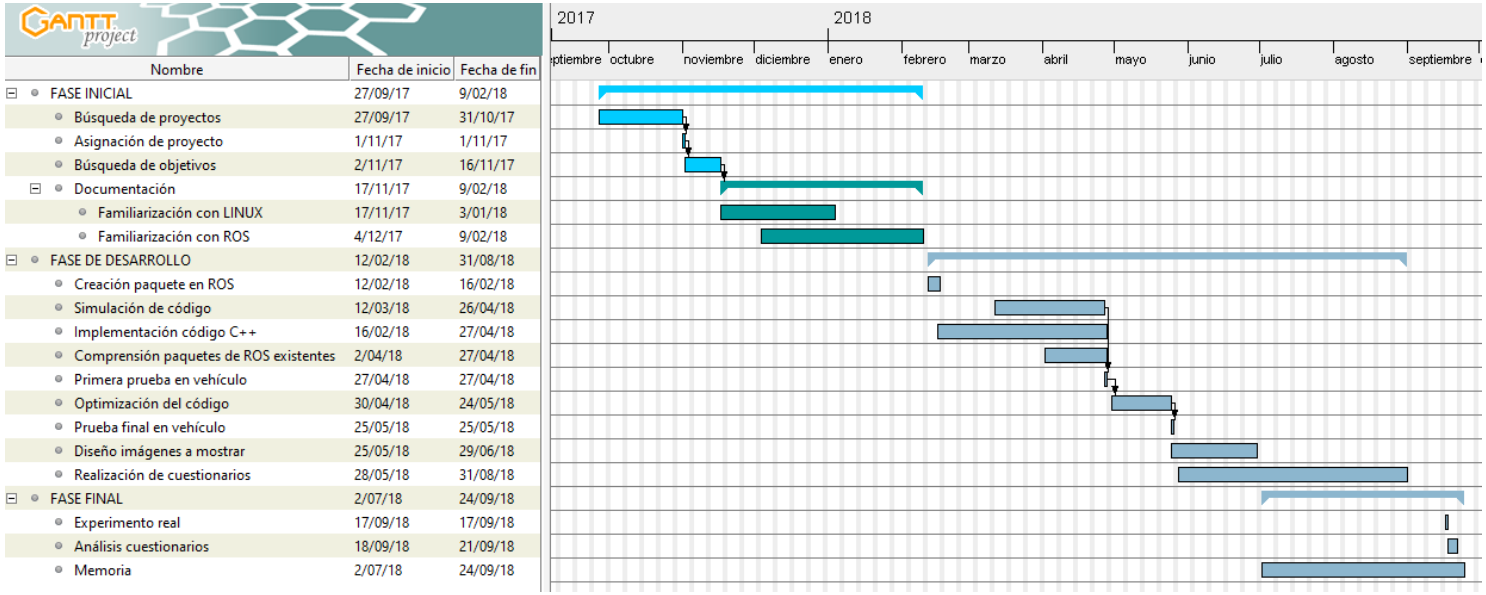
A medida que se vayan publicando más avances en el sector, los peatones tendrán más conocimientos al respecto y podrán identificar tanto la existencia de la pantalla como ser capaces de entender el mensaje que comunica.

6. PLANIFICACIÓN Y PRESUPUESTO

6.1. Planificación

Para la planificación se ha creado un Diagrama de Gantt en función de las etapas de desarrollo por las que ha ido pasando este proyecto (ver punto 4.1).

Se ha dividido el diagrama en las tres fases de desarrollo principales previamente mencionadas y explicadas, y se ha añadido en cada una de ellas las diferentes tareas realizas y el tiempo empleado por cada una de ellas, así como las fechas de su realización.



Nombre	Fecha de inicio	Fecha de fin
FASE INICIAL	27/09/17	9/02/18
Búsqueda de proyectos	27/09/17	31/10/17
Asignación de proyecto	1/11/17	1/11/17
Búsqueda de objetivos	2/11/17	16/11/17
Documentación	17/11/17	9/02/18
Familiarización con LINUX	17/11/17	3/01/18
Familiarización con ROS	4/12/17	9/02/18
FASE DE DESARROLLO	12/02/18	31/08/18
Creación paquete en ROS	12/02/18	16/02/18
Simulación de código	12/03/18	26/04/18
Implementación código C++	16/02/18	27/04/18
Comprensión paquetes de ROS existentes	2/04/18	27/04/18
Primera prueba en vehículo	27/04/18	27/04/18
Optimización del código	30/04/18	24/05/18
Prueba final en vehículo	25/05/18	25/05/18
Diseño imágenes a mostrar	25/05/18	29/06/18
Realización de cuestionarios	28/05/18	31/08/18
FASE FINAL	2/07/18	24/09/18
Experimento real	17/09/18	17/09/18
Análisis cuestionarios	18/09/18	21/09/18
Memoria	2/07/18	24/09/18

Ilustración 6.1: Diagrama de Gantt

6.2. Presupuesto

El cálculo del presupuesto del proyecto se ha separado en tres capítulos, alcanzando un total de **4863,88€**.

6.2.1. Capítulo 1: Hardware

Dentro del hardware se encuentran todos los elementos físicos necesarios para el desarrollo y posterior funcionamiento del proyecto.

Los costes de los elementos hardware han sido modificados de su precio de adquisición calculando la amortización de cada uno, es por ello por lo que el coste que aparece en el presupuesto es mucho menor. Es importante mencionar que la pantalla utilizada para mostrar las imágenes ha sido totalmente amortizada y por ello aparece un valor de 0€ en el presupuesto.

De acuerdo con eso, el presupuesto del hardware es el detallado a continuación:

CÓDIGO	UNIDAD	DESCRIPCIÓN	MEDICIÓN	PRECIO UNITARIO	PRECIO TOTAL
		CAPÍTULO 1 HARDWARE			
1.01	Ud	**VEHÍCULO AUTÓNOMO iCab 1** Carrito de golf modelo E-Z-GO, modificado eléctrica y mecánicamente para ser capaz de operar de forma autónoma.	1,00	239,58	239,58
1.02	Ud	**PANTALLA** Pantalla colocada en la parte frontal instalada para mostrar las imágenes previamente diseñadas al peatón.	1,00	0,00	0,00
1.03	Ud	**ORDENADOR PORTÁTIL LENOVO YOGA 520-14IKB** Ordenador portátil convertible 2 en 1 de Lenovo, pantalla táctil 14", procesador Intel Core i5, tarjeta gráfica dedicada NVIDIA GeForce GT 940MX.	1,00	161,80	161,80
				TOTAL	**401,38**

6.2.2. Capítulo 2: Software

CÓDIGO	UNIDAD	DESCRIPCIÓN	MEDICIÓN	PRECIO UNITARIO	PRECIO TOTAL
		CAPÍTULO 2 SOFTWARE			
2.01	Ud	**LINUX 16.04** Sistema operativo de libre licencia	8,00	0,00	0,00
2.02	Ud	**ROBOT OPERATING SYSTEM (ROS)** Framework utilizada para el desarrollo del software del vehículo autónomo. Plataforma de código abierto.	1,00	0,00	0,00
2.03	Ud	**Qt CREATOR** Plataforma IDE (Integrated Development Environment) utilizada para el desarrollo del código en C++	1,00	0,00	0,00
2.04	Ud	**OpenCV** Biblioteca libre de visión artificial.	1,00	0,00	0,00
				TOTAL	**0,00**

6.2.3. Capítulo 3: Personal

CÓDIGO	UNIDAD	DESCRIPCIÓN	MEDICIÓN	PRECIO UNITARIO	PRECIO TOTAL
		CAPÍTULO 3 PERSONAL			
3.01	Hora	**Dr. Fernando Fernández García** Ingeniero encargado de dirigir el proyecto	20,00	20,00	400,00
3.02	Hora	**Ahmed Hussein** Ingeniero encargado dirigir la parte específica del software realizado para el iCab 1	30,00	15,00	450,00
3.03	Hora	**Dr. Cristina Olaverri Monreal** Ingeniera encargada de dirigir la parte específica de la interacción vehículo autónomo-peatón	10,00	20,00	200,00
3.04	Hora	**María Huertas Martín** Estudiante de ingeniería encargada del desarrollo completo del proyecto	350,00	9,75	3412,50
				TOTAL	**4462,50**

Como se especifica en el "XVIII Convenio colectivo Nacional de empresas de ingeniería y oficinas de estudios técnicos" publicado en el BOE, a fecha de 18 de enero de 2017, el salario mínimo de un ingeniero es España es 17,544.24€ anuales con un máximo de 1800 horas trabajadas.

Con los anteriores datos, el salario mínimo en España es calculado como 17,544.24€/1800h siendo este igual a 9,75€.

6.2.4. Resumen

APARTADO	RESUMEN	IMPORTE
Capítulo 1	Hardware	401,38
Capítulo 2	Software	0,00
Capítulo 3	Personal	4462,50
	TOTAL	4863,88

El presupuesto total del proyecto asciende a **4863,88 €**.

7. CONCLUSIONES Y TRABAJOS FUTUROS

7.1. Conclusiones

En la memoria se presenta el desarrollo completo de una interfaz para implementar en el vehículo autónomo iCab 1. Dicha interfaz cumple con todos los objetivos necesarios para que sea capaz de existir una comunicación entre el vehículo autónomo y el peatón.

La implementación del código fue hecha en el lenguaje C++, permitiendo al paquete conectarse con el sistema previamente instalado en el vehículo. Además, gracias al topic creado, la interfaz es capaz de mantener contacto con el resto de las interfaces, pudiendo así recibir mensajes.

Se ha demostrado que el nodo configurado en el paquete es capaz de trabajar con los mensajes recibidos, pudiendo así emitir decisiones en forma de imagen que permitan la comunicación deseada con los peatones.

Las posteriores pruebas no reales realizadas en el vehículo han servido para optimizar el código. En un principio el paquete creado constaba de dos nodos en comunicación, el primero analizaba los mensajes enviados por otro paquete, y el segundo emitía una decisión. Gracias a dichas pruebas, se logró simplificar todo el paquete y código en un único nodo capaz de realizar todas las funciones sin ningún tipo de problemas.

A parte del código, en este proyecto se llevó a cabo la parte de diseño. Las imágenes que son mostradas a los peatones fueron diseñadas en función de las diferentes pruebas reales que se realizaron, así como con la intención de ser entendidas por el peatón. Causar una respuesta positiva en el peatón es lo que se buscaba, por lo que crear una serie de imágenes que fueran rápidamente comprendidas por él era primordial.

Para finalizar, se puso todo tanto el código como el diseño en conjunto, y se realizaron una serie de pruebas reales, donde los peatones no conocían previamente el sistema, y gracias a ello se obtuvieron una serie de resultados.

Dentro de la parte de pruebas, a pesar de realizarlas dentro de un único experimento donde se carecía de ambientación, se pueden concluir con buenos resultados por parte de los participantes. La gran mayoría de ellos coinciden en que la existencia de un sistema de comunicación entre vehículo sin conductor y peatón es positiva, incluso aquellos que no probaron la interfaz diseñada. Respecto a los participantes que no se percataron de la

existencia de una pantalla, incluso aquellos que no fueron capaces de interpretar el mensaje emitido, se puede reducir a una única razón. Se ha llegado a la conclusión de que eso ocurre debido a la falta de información y adoctrinamiento ciudadano ante la presencia de vehículos autónomos ya que, actualmente, por falta de regularizaciones legales, entre otras cosas, se ve la implantación autónoma en un futuro lejano.

A parte de alcanzarse el objetivo principal del proyecto, dentro de los objetivos personales, se consiguieron adquirir nuevos conocimientos en el área. Se aprendió a utilizar ROS, debido a que no se partía de una base previa, y a medida que se avanzaba con el proyecto estos fueron aumentando.

También hay que destacar que, aunque en el lenguaje de programación C++ se partía de una base académica, gracias al proyecto se aprendió a orientar esos conocimientos a un área desconocida con todo lo que ello supone. Se aprendió también cómo trabajar con nuevas librerías, así como un uso diferente en punteros.

Para concluir con los objetivos personales estaría el aprendizaje a otro nivel del sistema operativo Linux, que como se ha ido comentando a lo largo de la memoria, fue el utilizado para el desarrollo del proyecto.

En resumen, en la memoria se presenta un proyecto con todas sus fases, implementación, diseño y resultados, y con todos los objetivos iniciales cumplidos. Se ha creado una interfaz que permite al vehículo autónomo ser capaz de comunicarse con los peatones, y ayudarles a la hora de tomar la decisión de cruzar la vía.

7.2. Desarrollos futuros

A continuación, se van a mencionar algunos de los posibles desarrollos futuros que mejorarían la interacción entre el vehículo autónomo y el peatón.

El primero sería ampliar el rango de posible peatón con el que interactuar. Es decir, nuestro sistema únicamente es capaz de mostrar imágenes, lo cual deja fuera de entendimiento a personas con problemas de vista, incluso niños o personas con discapacidades para el entendimiento. Por lo que para mejorar eso se podrían añadir avisos sonoros, por ejemplo, emitidos por el propio vehículo. También sería buena idea complementar todo el sistema con sistemas de luces. Multitud de compañías están

barajando la idea de añadir luces al coche con diferentes frecuencias de parpadeo y colores, separando así los estados por los que pasa el vehículo.

Esto último, los diferentes estados, también es otra de las posibles mejoras que se están implementando en los vehículos autónomos. La principal idea es mostrar al peatón un mensaje que avise de cuándo es detectado, pero también se podrían emitir diferentes estados. Por ejemplo, avisar de cuando es detectado, cuando está empezando a perder velocidad, cuando es completamente seguro cruzar, o incluso mostrar periodos de velocidad por los que va pasando antes de detenerse.

También nos encontramos con que hay diferentes vías de que se le comunique al peatón que fue detectado. En el proyecto solo se abordó una de ellas, que sería un mensaje que se encuentra dentro del vehículo y alerta de ello. Por lo que, los siguientes desarrollos que se pueden aplicar sería ampliar esto a otras vías. Todas ellas son complementarías a la actual, y serían las siguientes [46]:

- Vehículo + carretera: utilizar la carretera como mensajera. Bien sea colocando luces en ella u otro tipo de señales luminosas.
- Vehículo + peatón: se le puede avisar al peatón por medio de aplicaciones de móviles. Estas solo tendrían que emitir un mensaje, que no tendría por qué ser únicamente texto. Una de las posibles soluciones sería el envío de mensajes sonoros, o vibraciones al móvil, enviadas por el coche, y que facilitarían el entendimiento al peatón sin necesidad de estar pendiente del móvil.
- Vehículo + carretera + peatón: un conjunto de todas ellas sería el sistema ideal, ya que un fallo de una interfaz siempre se puede complementar con la información de las otras dos. Este sistema incluso se podría permitir el fallo de dos de ellas, ya que habría una tercera operativa.

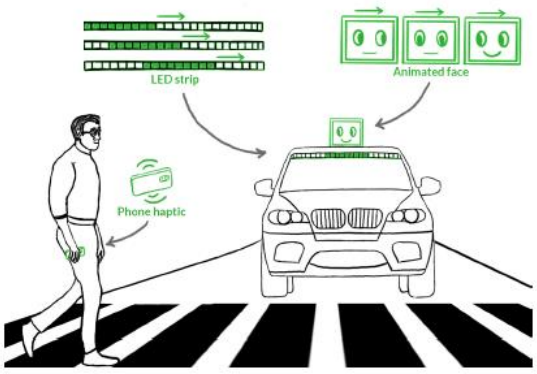


Ilustración 7.1: Diseño de posibles desarrollos futuros [47]

Y, por último, en la parte estética del diseño, sería conveniente mejorar la pantalla en tamaño y luminosidad, para evitar así reflejos del sol que impidan ver con claridad el mensaje que se muestra.

BIBLIOGRAFÍA

[1] A. Barredo y L. del Valle Hernández, «programarfacil.com,» [En línea]. Available: https://programarfacil.com/podcast/coche-autonomo-estado-del-arte/.

[2] M. Fallon, Self-Driving Cars: The New Way Forward, 2018.

[3] H. Reese, «TechRepublic,» [En línea]. Available: https://www.techrepublic.com/article/our-autonomous-future-how-driverless-cars-will-be-the-first-robots-we-learn-to-trust/.

[4] L. Dormehl y S. Edelstein, «Digital Trends,» 18 1 2018. [En línea]. Available: https://www.digitaltrends.com/cars/history-of-self-driving-cars-milestones/.

[5] Encyclopedia Britannica. Inc, «Artificial Intelligence,» de *Technology: Britannica Illustrated Science Library*, 2008, p. 79.

[6] Google, «Waymo,» [En línea]. Available: https://waymo.com/journey/.

[7] S. Edelstein, «Can A.I. make self-driving cars a reality? Waymo reveals the future,» Digital Trends, 5 8 2018. [En línea]. Available: https://www.digitaltrends.com/cars/google-io-2018-waymo-recap/.

[8] BMW, «Autonomous Driving,» [En línea]. Available: https://www.bmw.com/en/automotive-life/autonomous-driving.html.

[9] B. C. Zanchin, R. Adamshuk y M. M. Santos, «On the instrumentation and classification of autonomous cars,» de *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Banff, AB, Canada, 2017.

[10] Intel, «Autonomous Driving – Hands on the Wheel or No Wheel at All,» 11 2018. [En línea]. Available: https://newsroom.intel.com/news/autonomous-driving-hands-wheel-no-wheel-all/.

[11] M. Adalid, «Futurism,» [En línea]. Available: https://futurism.com/images/laws-and-ethics-for-autonomous-cars/.

[12] J. C. Sánchez y C. G. Fernández-Vallejo, 11 8 2017. [En línea]. Available: https://www.technologyreview.es/s/8801/el-problema-de-la-conduccion-autonoma-es-basicamente-legal-no-tecnologico.

[13] A. I. Fraga, «TICbeat,» 19 7 2015. [En línea]. Available: http://www.ticbeat.com/innovacion/los-coches-autonomos-avance-retroceso-para-el-hombre/.

[14] A. N.-T. Stock, «TICbeat,» 8 1 2018. [En línea]. Available: http://www.ticbeat.com/innovacion/la-dgt-se-asocia-con-mobileye-para-la-llegada-de-los-vehiculos-autonomos/.

[15] mobileye, «mobileye,» 22 6 2017. [En línea]. Available: https://www.mobileye.com/es-es/2017/06/22/la-dgt-comienza-apostar-por-el-uso-generalizado-de-adas/.

[16] T. Peng y M. Sarazen. [En línea]. Available: https://medium.com/syncedreview/global-survey-of-autonomous-vehicle-regulations-6b8608f205f9.

[17] N. López, «Autobild.es,» 13 12 2017. [En línea]. Available: https://www.autobild.es/reportajes/legislacion-coche-autonomo-espana-queda-mucho-hacer-179660.

[18] R. C. Arkin, «Ethics and Autonomous Systems: Perils and Promises,» *Proceedings of the IEEE*, vol. 104, nº 10, pp. 1779-1781, 2016.

[19] A. Soler, «Los coches autónomos están perdiendo terreno en Europa,» Híbridos y Eléctricos, 7 8 2018. [En línea].

[20] DGT, «En 2017, 1.200 fallecidos,» [En línea]. Available: http://revista.dgt.es/es/noticias/nacional/2018/01ENERO/0103-Presentacion-balance-accidentes-2017.shtml#.W6apdWgzbIU.

[21] D. M. Gómez, P. M. Plaza, A. Hussein, A. d. l. E. Hueso y J. M. A. Moreno, «ROS-based Architecture for Autonomous Intelligent Campus Automobile (iCab),» Madrid.

[22] D. M. Gómez, P. M. Plaza, A. Hussein, A. d. l. E. Hueso y J. M. A. Moreno, «Complete ROS-based Architecture for Intelligent Vehicles,» Leganés, Madrid.

[23] «UNIX Tutorial for Beginners,» [En línea]. Available: http://www.ee.surrey.ac.uk/Teaching/Unix/.

[24] Erle Robotics, «Introducción de ROS,» [En línea]. Available: https://erlerobotics.com/blog/ros-introduction-es/#What_is_ros.

[25] Universidad de Alicante, «Visión artificial y robótica,» [En línea]. Available: https://moodle2015-16.ua.es/moodle/mod/book/view.php?id=82546&chapterid=2346.

[26] ROS, «Creating a ROS Package,» [En línea]. Available: http://wiki.ros.org/ROS/Tutorials/CreatingPackage.

[27] A. Mahtani, L. Sanchez, E. Fernández y A. Martínez, Effective Robotics Programming with ROS, Birmingham, UK: Packt Publishing Ltd, 2016.

[28] ROS, [En línea]. Available: http://wiki.ros.org/catkin/CMakeLists.txt.

[29] ROS, «ROS Conceps,» [En línea]. Available: http://wiki.ros.org/ROS/Concepts#ROS_Computation_Graph_Level.

[30] Ł. Mitka. [En línea]. Available: https://husarion.com/tutorials/ros-tutorials/2-creating-nodes/#2-creating-nodes-workspace-setup.

[31] ROS, «Understanding ROS Topics,» [En línea]. Available: http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics.

[32] VAKOMS, «Why You Should Choose Qt For Cross-Platform App Development,» [En línea]. Available: https://www.upwork.com/hiring/for-clients/qt-cross-platform-app-development/.

[33] J. D. L. Rivas, «CREATING ROBOTS,» [En línea]. Available: https://creatingrobots.wordpress.com/qt-creator-ros/.

[34] A. Rasouli, I. Kotseruba y J. K. Tsotsos, «Agreeing To Cross: How Drivers and Pedestrians Communicate,» 2017.

[35] D. Tripolone, «Land Rover develops virtual eyes to get others to trust driverless cars,» 29 8 2018. [En línea]. Available: https://www.news.com.au/technology/innovation/motoring/hitech/land-rover-develops-virtual-eyes-to-get-others-to-trust-driverless-cars/news-story/5c86ce7b1c5edd04be85eb7a6af7c7fe.

[36] R. S. Aouf, «Jaguar Land Rover's prototype driverless car makes eye contact with pedestrians,» 4 9 2018. [En línea]. Available: https://www.dezeen.com/2018/09/04/jaguar-land-rovers-prototype-driverless-car-makes-eye-contact-pedestrians-transport/.

[37] J. F. B. G. T. S. Morgan Quigley, «ros.h File Reference,» 7 3 2017. [En línea]. Available: http://docs.ros.org/jade/api/roscpp/html/ros_8h.html.

[38] K. C. J. L. Morgan Quigley. [En línea]. Available: http://wiki.ros.org/std_msgs.

[39] The Open Group, [En línea]. Available: http://pubs.opengroup.org/onlinepubs/7908799/xsh/string.h.html.

[40] OpenCV, [En línea]. Available: https://opencv.org/.

[41] W. Woodall, 3 11 2015. [En línea]. Available: http://wiki.ros.org/roscpp/Overview/Initialization%20and%20Shutdown.

[42] J. Faust, Open Source Robotics Foundation , 17 11 2009. [En línea]. Available: http://wiki.ros.org/roscpp/Overview/NodeHandles.

[43] ROS, «Writing a Simple Publisher and Subscriber (C++),» Open Source Robotics Foundation, [En línea]. Available: http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29.

[44] ROS, «Callbacks and Spinning,» [En línea]. Available: http://wiki.ros.org/roscpp/Overview/Callbacks%20and%20Spinning.

[45] A. Pillai, «Virtual Reality based Study to Analyse Pedestrian Attitude towards Autonomous Vehicles,» Espoo, Finland, 2017.

[46] K. Mahadevan, S. Somanath y E. Sharlin, «Communicating Awareness and Intent in Autonomous Vehicle-Pedestrian Interaction,» 2018.

[47] K. Mahadevan, S. Somanath y E. Sharlin, «Communicating Awareness and Intent in Autonomous Vehicle-Pedestrian Interaction,» de *CHI Conference on Human Factors in Computing Systems*, Montreal QC, Canada, 2018.